Universidad Zaragoza
1542

Trabajo Fin de Grado

Localizador genérico de galerías en un túnel a partir de un mapa topológico previo.

Generic gallery locator in a tunnel from a previous topologic map.

Autor:

Carolina Tello Vicente

Director:

Luis Montano Gella

Escuela de Ingeniería y Arquitectura de Zaragoza

2019

Localizador genérico de galerías en un túnel a partir de un mapa topológico previo.

Generic gallery locator in a tunnel from a previous topologic map.

RESUMEN:

El objeto de este proyecto es la localización absoluta de un robot móvil en un túnel a partir de un mapa topológico en el que están localizadas las galerías laterales, como lugares relevantes a reconocer por el sistema de percepción embarcado. Estas galerías, que pueden estar tanto a derecha como a izquierda, se reconocerán en ambos sentidos de ida y vuelta. El método genérico de reconocimiento y localización de galerías desarrollado será aplicable a cualquier túnel con galerías laterales. No es necesario disponer y manejar un mapa geométrico previo, en los que el mantenimiento y actualización del mapa, y la localización del robot en él es un proceso costoso en tiempo de procesamiento y en ocupación de memoria. Esta localización se integrará con medidas odométricas y de "scan matching" en un filtro de Kalman de estimación para tener una localización continua más precisa.

En escenarios extensos como los considerados en este trabajo, el error de localización absoluta del robot crece enormemente si no hay características geométricas reconocibles entre lugares relevantes, en nuestro caso las galerías, en mayor medida si no se dispone de un mapa geométrico denso, que es la hipótesis de este trabajo.

Para la localización se ha desarrollado un algoritmo que, utilizando los datos del escáner láser frontal (LIDAR) embarcado en el robot, genere el mapa basado en rectas, en el que se detectarán las galerías. Tras la detección de galerías se tomará como medida para la localización un punto representativo de las mismas fácilmente localizable en el mapa. La localización absoluta de este punto será conocida en el mapa topológico del que se parte. Ello permitirá realizar una corrección de la localización absoluta del robot en el túnel cada vez que se detecte una galería, reduciendo el error acumulado hasta esa posición.

Estos algoritmos se han desarrollado para su puesta a punto inicialmente en MATLAB, evaluándose los resultados con datos reales del antiguo túnel ferroviario de Somport. Tras ello, se han integrado los algoritmos de reconocimiento y localización en las plataformas de ROS *(Robot Operating System)*, un entorno de desarrollo para robótica, y Gazebo, un simulador. Se crearán distintos escenarios para validar la capacidad de reconocimiento de galerías en diferentes túneles y por tanto la generalidad del método de localización propuesto. Con estas herramientas se pondrán a punto los algoritmos desarrollados y el ajuste de parámetros final previamente a la implementación en un robot real en el futuro.

Tabla de contenido

Índice de figuras

1. Introducción

En este capítulo se explicarán los aspectos principales de la memoria: sus objetivos, alcance y contenido. Además se explicará el trabajo previo del que se ha partido para elaborar este proyecto.

1.1. Contexto y alcance

Este proyecto se ha basado en unas premisas iniciales en las que se pretende disminuir el costo en tiempo de procesamiento y ocupación de memoria, por ello se utilizan mapas topológicos en donde se encuentra la información más relevante, sin tener que mantener ni actualizar un mapa geométrico, que es más bien denso, ni la localización en él.

Por el contrario, el hecho de que los escenarios de este trabajo son extensos, el error de localización absoluta del robot puede crecer enormemente si no hay características geométricas reconocibles entre lugares relevantes. Habitualmente el problema de la localización del robot se resuelve, bien utilizando la información odométrica a partir de los sensores de velocidad de las ruedas, la información de un sensor inercial (IMU), utilizando técnicas de "scan matching", técnicas SLAM, o la integración de algunas o todas ellas. La utilización de odometría e IMU conduce a un incremento continuo del error de localización, que debe corregirse con las otras técnicas. Pero estas, a su vez, necesitan que exista un mapa geométrico previo conocido para reducir el error de localización. Estos mapas, si el escenario es extenso como en el caso que se trata en este proyecto, necesitan una gran ocupación en memoria y coste elevado de procesamiento, que dificulta su utilización en tiempo real, mientras el robot navega. Además, si no existen características geométricas reconocibles, por ejemplo paredes planas o mala iluminación habitual en túneles o galerías subterráneas, estas técnicas tampoco permiten reducir el error de localización.

Una alternativa es la de utilizar mapas topológicos, en los que solamente se incluyen características topológicas relevantes reconocibles, es decir con información semántica, cuya localización se conoce con una cierta precisión. Un ejemplo de ellas son las galerías laterales en túneles o cruces de galerías en minas o laberintos de galerías. Disponer de un mapa topológico con este tipo de características presenta ventajas:

- No es necesario para la localización un mapa geométrico denso del escenario, con lo que la ocupación en memoria y, sobre todo el coste de procesamiento se reduce considerablemente.
- El reconocimiento de estas características permite reducir esporádicamente, cuando son detectadas y reconocidas, el error de localización que incrementa cuando se utiliza odometría o IMU exclusivamente.

Pero también presenta inconvenientes:

- Es necesario desarrollar un sistema de reconocimiento en tiempo real de estas características, normalmente previo aprendizaje de las mismas.
- Hay que integrar el sistema de localización básico con la localización de estas características.

Precisamente abordar estos dos problemas es la contribución de este trabajo. Se van a considerar escenarios tipo túnel con galerías laterales, del que se conoce un mapa topológico con la localización de estas galerías en él. Las características a reconocer son estas galerías, que pueden ser de paredes irregulares, de diferentes anchuras, y en diferentes orientaciones respecto al corredor central del túnel. Es decir, hay que desarrollar una técnica que sea robusta ante esta posible variación de sus características geométricas.

Este tipo de escenarios tiene un interés creciente en la comunidad robótica, por las múltiples aplicaciones en las que la robotización puede contribuir a la realización de trabajos en entornos hostiles para los humanos. Por ejemplo, túneles en construcción, inspección en galerías de instalaciones urbanas, reconstrucción topográfica de cuevas, logística en minas, revisiones periódicas en los túneles, en consecuencia a la directiva que creó la Unión Europea 2004/54/CE del Parlamento europeo y del Consejo sobre requisitos mínimos de seguridad para túneles de la red transeuropea de carreteras, para prevenir accidentes y ejecutar las reparaciones necesarias entre otros. Estas son solo algunas aplicaciones y escenarios de interés, por mencionar algunas en las que se están utilizando actualmente sistemas robotizados.

1.2. Objetivos

El objetivo principal de este proyecto es el cálculo de la localización absoluta de un robot móvil en un túnel a partir del reconocimiento de galerías tanto a izquierda como a derecha, en ambos sentidos de ida y vuelta, permitiendo así reducir el error de localización absoluta. El método genérico de reconocimiento y localización de galerías desarrollado será aplicable a cualquier túnel con galerías laterales.

Los objetivos específicos que se plantean alcanzar en este proyecto son los siguientes:

- Revisión del algoritmo preliminar desarrollado en el Grupo de Robótica de la Universidad de Zaragoza para la identificación de galerías, realización de posibles mejoras y ajuste de parámetros para mejorar la identificación de estas características.
- Localización de galerías y cálculo de la localización absoluta del robot móvil a partir de los datos disponibles del antiguo túnel ferroviario de Somport utilizando MATLAB, escenario habitual de experimentación del Grupo de Robótica.
- Integración en la plataforma software y de simulación ROS-Gazebo de los algoritmos de reconocimiento de galerías y localización creados en MATLAB, así como la puesta a punto fina en ellos.
- Creación de nuevos escenarios con las herramientas proporcionadas por el simulador Gazebo.
- Validación del método creado en los diferentes escenarios y ajuste final.

1.3. Trabajo previo

Para la recogida de datos en el túnel de Somport, se han utilizado dos tipos de vehículos:

- Un vehículo todo-terreno Galloper, equipado con dos encoders incrementales SICK DSF60 y un LIDAR SICK LMS200, se muestra imagen en la Figura 1:

Figura 1. Galloper equipado con dos encoders SICK DSF60 y un LIDAR SICK LMS200

Estos dos encoders incrementales pueden ser programados si se requiere tanto la señal de salida como el pulso cero. El láser frontal LIDAR usado, tiene un ángulo de abertura de 180º, con una resolución angular de 1º y un rango máximo de alcance de hasta 80 metros mientras que el rango máximo con reflectividad al 10% es de 10 metros.

- La plataforma Robucar equipada con dos sensores láser frontales SICK LMS200, uno situado horizontalmente para detección de obstáculos frontales, otro orientado hacia el suelo para detectar obstáculos sobre el mismo; y otros dos sensores láser traseros SICK LMS291/200 con una configuración igual que los delanteros. Se muestra imagen en la Figura 2:

Figura 2. Robucar equipado con dos sensores láser frontales SICK LMS200 y dos sensores láser traseros SICK LMS291/200.

La plataforma Robucar utiliza una IMU, *Intertial Measurement Unit*, para conocer su posición, compuesta por una serie de acelerómetros y giróscopos que miden cuánto se mueve y gira el robot. Los dos sensores láser frontales son los mismos

utilizados en el Galloper, mientras que uno de los traseros es SICK LMS291 y tiene un ángulo de abertura de 180º, con una resolución angular de 1º y un rango máximo de alcance de hasta 80 metros mientras que el rango máximo con reflectividad al 10% es de 30 metros.

Para las simulaciones en ROS se ha utilizado el modelo de un Pioneer P3-AT equipado con un LIDAR SICK LMS200, como muestra la Figura 3:

Figura 3. Pioneer P3-AT (MobileRobots) equipado con LIDAR SICK LMS200.

Esta plataforma cuenta con encoders de cuadratura óptica de alta resolución. El sensor láser LIDAR es el utilizado también por el Galloper y Robucar.

En la Figura 4 se muestra una parte pequeña del túnel central con 3 galerías laterales, obtenida con un escáner laser embarcado en un robot. Como se puede observar las galerías son diferentes y están en diferentes orientaciones. El algoritmo ha de reconocerlas independientemente de su forma y orientación, sin tener esta descripción correspondiente a un mapa geométrico. Es decir, sólo estará disponible información topológica, la ubicación (posición de su entrada) de una galería genérica en el túnel. Cabe mencionar que el túnel tiene 7.7 km de longitud, uniendo España y Francia. Es decir es un escenario extenso con múltiples galerías laterales.

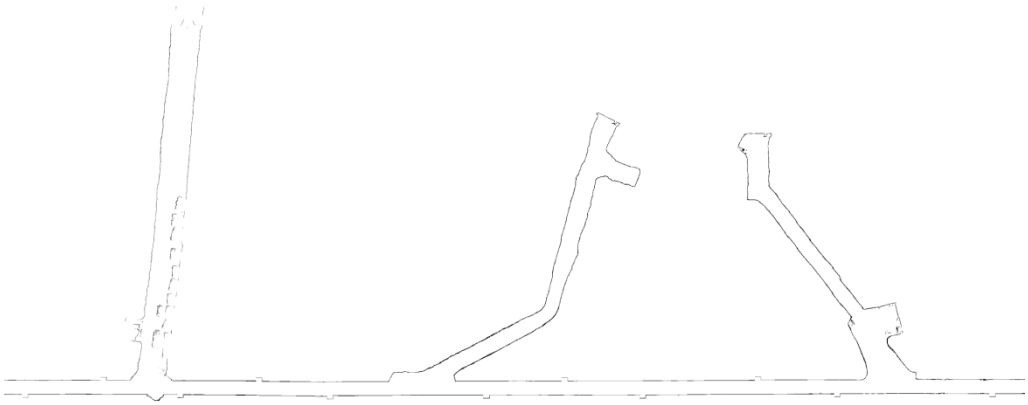


Figura 4. Tres galerías del túnel de Somport.

1.4. Contenido de la memoria

La memoria de este proyecto está estructurada de la siguiente forma:

- **Capítulo 1. Introducción.** En este apartado se indican el contexto, el alcance, los objetivos principales y el trabajo previo del proyecto.

- **Capítulo 2. Reconocimiento de galerías laterales en un túnel.** En este apartado se explica el algoritmo base inicial de partida del reconocimiento de galerías y las mejoras propuestas y aplicadas en él.

- **Capítulo 3. Localización absoluta del robot en un túnel.** En este apartado se explica el cálculo de la posición del robot en base al punto representativo de la galería y la corrección de la localización aplicada.

- **Capítulo 4. Resultados obtenidos con los datos del túnel de Somport.** En este apartado se expone y explica los resultados gráficos principales obtenidos de la simulación con los datos del antiguo túnel ferroviario de Somport.

- **Capítulo 5. Implementación en ROS.** En este apartado se explica la integración de los algoritmos en ROS junto con las diferencias significativas de trabajo entre ROS y MATLAB.

- **Capítulo 6. Creación de escenarios en Gazebo.** En este apartado se habla sobre la herramienta disponible en Gazebo para construir elementos de túnel y creación y visualización de escenarios.

- **Capítulo 7. Resultados finales.** En este capítulo se exponen los resultados finales con un mapa representativo creado en Gazebo.

- **Capítulo 8. Conclusiones.** En este capítulo se finaliza la memoria presentando las conclusiones finales del proyecto, realizando un análisis de los objetivos establecidos y su consecución y/o sus posibles defectos.

2. Reconocimiento de galerías laterales en un túnel

En este trabajo se ha partido de un algoritmo preliminar desarrollado en el grupo de Robótica. Se ha evaluado y se han introducido un gran número de mejoras que hacen más general y robusto el método.

En este capítulo se procede a explicar el algoritmo base que detecta las galerías laterales del túnel, siendo este el punto de partida del proyecto. Este detector reconoce galerías laterales en un túnel genérico, es decir, que las galerías corresponden a parámetros genéricos definidos para un escenario y constituyen los descriptores de cualquier galería. Estos parámetros descriptores pueden reconocer galerías dentro de un amplio rango de posibles valores y por tanto no se necesita un previo aprendizaje de las galerías específicas de un túnel.

Los parámetros iniciales considerados en este detector son:

- La apertura mínima de la galería para considerar una discontinuidad y segmentar el mapa sensorial obtenido.
- El mínimo número de puntos soporte para el segundo segmento constitutivo de la galería. Los puntos soporte son aquellos considerados para el reconocimiento como tal de una pared lateral de galería. Al ser habitualmente paredes irregulares (nube de puntos), se aproximan a segmentos para hacer el reconocimiento y calcular la localización
- La distancia mínima entre segmentos soporte para validar una galería. Los segmentos soporte son aquellos constitutivos de la galería.
- El número mínimo de reconocimientos consecutivos de una galería potencial para validarla como galería real. Esta información de persistencia es necesaria para eliminar espúreos, al ser escenarios y medidas reales sometidas a ruido.
- La mínima desorientación del robot respecto al eje de avance del sistema de referencia del túnel. El eje del túnel se considerará como eje X del sistema de referencia absoluta definida para todo el túnel. Por ello la desorientación respecto a este eje permite definir la orientación absoluta del robot en todo momento.
- El ángulo frontal de visión del sensor, utilizado para el filtrado de potenciales puntos del suelo encontrados en el avance (filtrado de puntos no relevantes). La detección de puntos en el suelo por el láser frontal, debido a oscilaciones de la plataforma al ser un terreno irregular, puede perturbar el reconocimiento de galerías, por lo que deben ser eliminados antes del procesamiento.

Todos estos parámetros tienen una interpretación geométrica por lo que se puede adaptar a otros túneles de galerías de características diferentes fácilmente.

El método, antes de explicar cada apartado en profundidad, funciona de la siguiente manera:

En cada posición del robot se obtienen los puntos del barrido láser. Estos puntos se segmentan, agrupando en segmentos un número mínimo de puntos próximos entre sí, segmentos que, en caso de haber una galería, corresponden a las paredes que establecen los límites de la galería (ver Figura 5 paredes 3 y 4). Aquellas parejas de segmentos consecutivos que cumplen la condición de distancia mínima (apertura de la galería)

entre los extremos de ambos segmentos, se establecen como soporte de una potencial galería. Cuando se detectan en un mínimo de posiciones del robot consecutivas una potencial galería, se valida como galería real. Una vez validada, se actualiza la localización odométrica del robot con el valor conocido en el mapa topológico de la situación de la galería en la referencia absoluta.

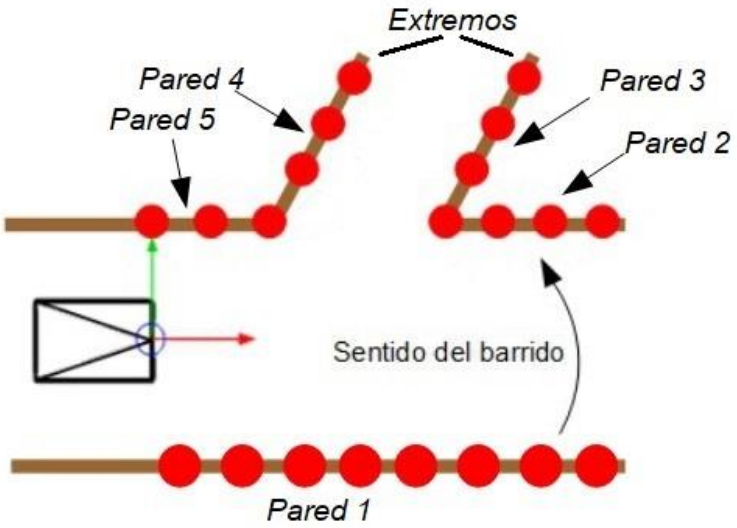


Figura 5. Representación de una galería.

El cálculo de la localización del robot, posición y orientación (x,y,θ), en la referencia absoluta se realiza integrando 3 informaciones diferentes:

- La X a partir de la odometría cuando no hay galerías laterales y a partir de la posición X en el mapa topológico cuando se detecta y reconoce una galería.
- La Y a partir de la distancia a las paredes laterales, o de una de ellas cuando en la otra hay una galería o deformidad importante, obtenida de la información del escáner láser.
- La θ a partir de la orientación de las dos paredes paralelas laterales del túnel, o de una de ellas, cuando no son paralelas, por existir una galería o una deformidad en una de ellas.

Para llevar a cabo este método se ha creado un algoritmo en MATLAB, en el que se pueden distinguir los siguientes pasos:

- Carga de datos:

En MATLAB, se ha trabajado de manera *offline*, es decir, con unos datos recopilados anteriormente, procedentes del escáner láser frontal y de la localización obtenida por medidas odométricas y de "*scan matching*".

Una vez cargados los datos en MATLAB se procede al cálculo de la posición absoluta y relativa del robot durante el movimiento a lo largo del túnel. Las posiciones relativas se calculan previamente para reutilizarlas cada vez que se reconozca una galería y se reinicialice con su valor correcto la localización. Conocidas las posiciones absolutas del paso anterior y actual de la odometría original ($^{O}T_{P1}$ y $^{O}T_{P2}$ respectivamente), se procede al cálculo de la posición relativa mediante

transformaciones, siendo *hom* la función que transforma un vector de localización (x,y,θ) en su correspondiente matriz homogénea, *inv* referenciando al cálculo de la matriz inversa y *loc* la función que transforma una matriz homogénea en vector de localización:

$$^{O}T_{P1} = \text{hom}([X_{ant}, Y_{ant}, \theta_{ant}])$$
$$^{O}T_{P2} = \text{hom}([X_{act}, Y_{act}, \theta_{act}])$$
$$^{P1}T_{O} = inv(^{O}T_{P1})$$
$$^{P1}T_{P2} = ^{P1}T_{O} * ^{O}T_{P2}$$
$$^{P1}x_{P2} = loc(^{P1}T_{P2})$$

Se realiza la inicialización de un parámetro general del algoritmo, el sentido del robot, ida o vuelta, que permite decidir si las galerías a encontrar en el mapa topológico están a la derecha o a la izquierda en el sentido del movimiento.

- Tratamiento de cada barrido láser en cada posición del robot:

1. Filtración de puntos no relevantes:

En cada posición del robot se obtienen los puntos del barrido láser frontal (180º), realizado de derecha a izquierda, de los cuales para eliminar información innecesaria, se filtran los puntos alejados más allá de 25 metros, considerados como puntos no relevantes. De estos puntos filtrados, dados en coordenadas polares, se obtienen también sus coordenadas cartesianas para facilitar los siguientes procesos.

2. Filtración de puntos que pertenecen al suelo:

Es posible que el láser situado en el robot cabecee proyectándose en el suelo y aparezcan puntos que puedan empeorar la calidad de la segmentación y las rectas. Estos puntos se eliminan para que no perturben el reconocimiento de galerías, que corresponderán a puntos pertenecientes exclusivamente a la pared.

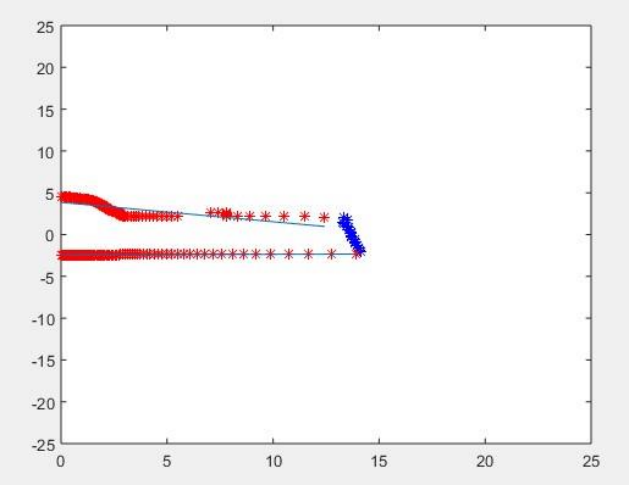


Figura 6. En azul, puntos filtrados pertenecientes al suelo.

Existen dos casos para poder filtrar el suelo mostrados en el siguiente pseudocógido:

$$if\ Y \approx 0\ and\ X > width:$$

$$Filtro\ 1: Y \approx 0 \pm \Delta\theta$$

$$else\ if\ Y \approx 0\ and\ Y_{all_{points}} < width:$$

$$Filtro\ 2: Y \approx 0 \pm \Delta\theta$$

$$else:$$

$$No\ existen\ puntos\ en\ suelo$$

En caso de que existan puntos cuyo valor en *Y* sea aproximadamente 0 y estén en un valor de *X* mayor que el de la anchura del túnel, se aplica el Filtro 1: se filtrarán esos puntos incluidos los que estén en un margen de θ seleccionado en el parámetro ajustable establecido para ello. Si el robot cabecea de tal forma que todavía quedan puntos pertenecientes al suelo, con valores de *X* menores a la anchura del túnel, se comprueba si los valores de *Y* de todos los puntos están dentro de la anchura del túnel. Si ese es el caso, se aplica el Filtro 2, filtrando esos puntos con valor de *Y* aproximadamente 0 incluidos los que estén en un margen de θ.

En las Figuras 7 y 9 se muestran el filtrado de los puntos que pertenecen al suelo cuando el movimiento está aproximadamente alineado con el eje del túnel y en las Figuras 8 y 10 se muestra que no se realiza el filtrado porque son puntos correspondientes a la pared cuando el robot está más orientado hacia ella. Es importante hacer esta distinción para no eliminar puntos de pared en los que se pudiera reconocer una galería. En la Figura 6 se muestra como se filtran los puntos en el algoritmo.

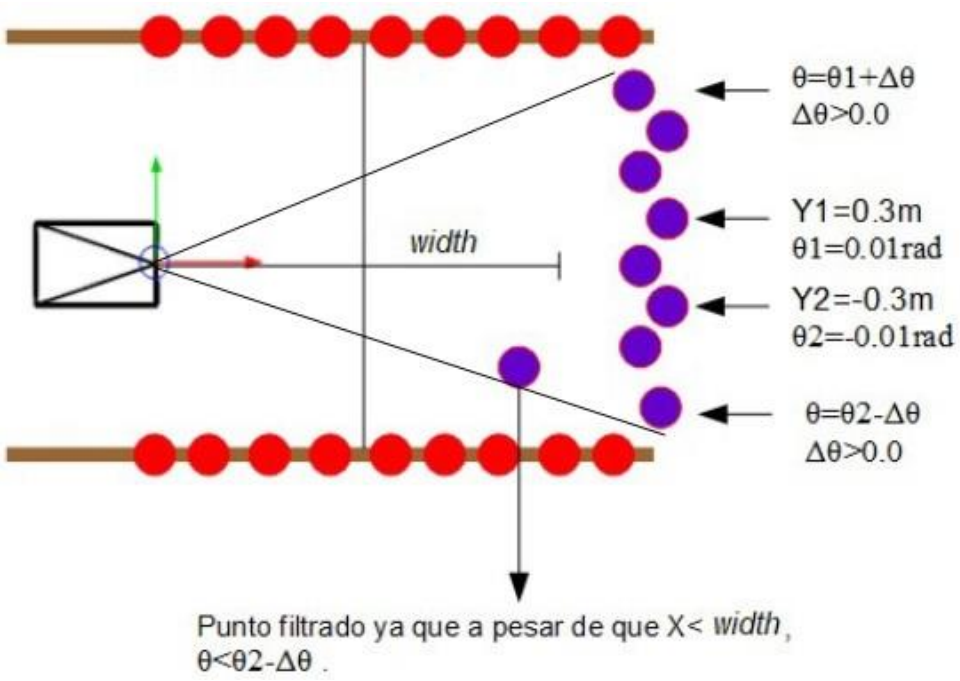


Figura 7. Filtro 1: puntos morados filtrados pertenecientes al suelo.

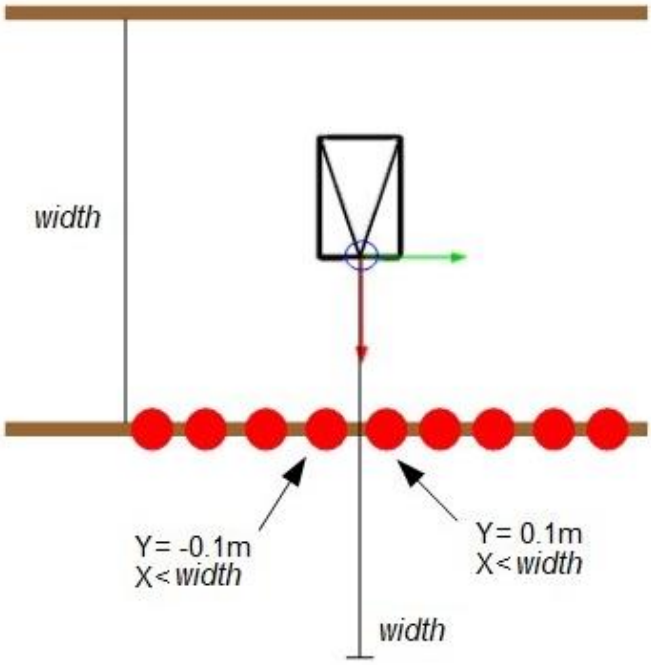


Figura 8. Filtro 1: No filtra puntos pertenecientes a la pared.

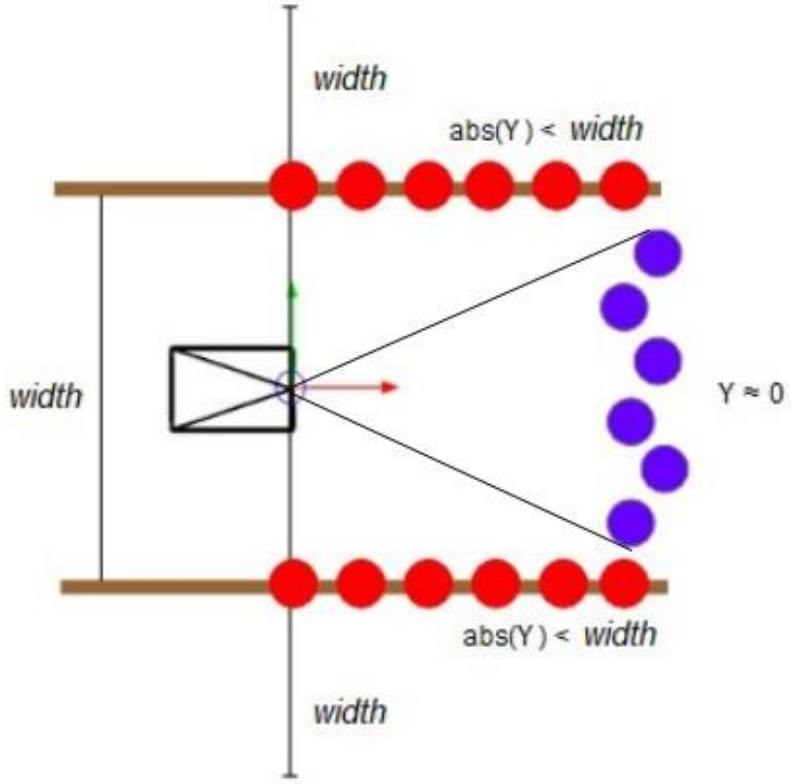


Figura 9. Filtro 2: puntos morados filtrados pertenecientes al suelo.

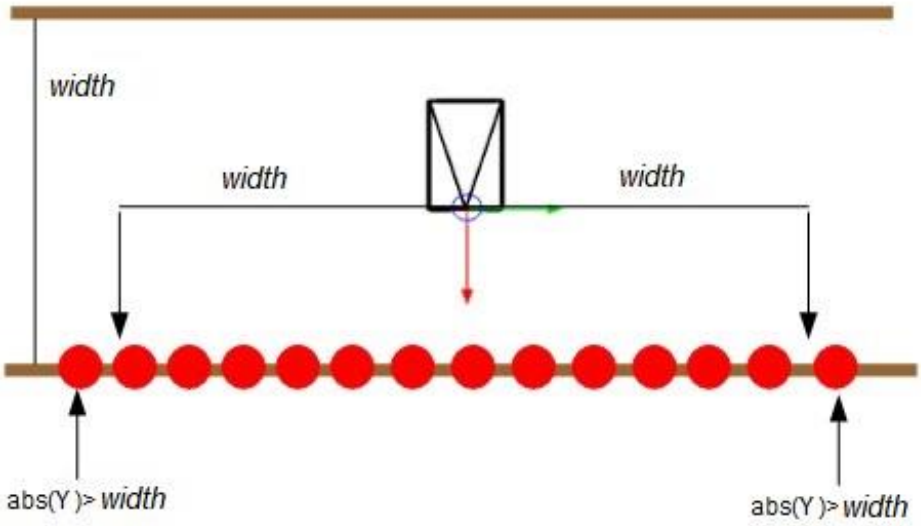


Figura 10. Filtro 2: No filtra puntos pertenecientes a la pared.

3. Segmentación de los puntos:

Una vez obtenidos los puntos se procede a su segmentación. Se agrupan puntos consecutivos en un segmento y se genera un nuevo segmento cuando ocurre alguno de estos sucesos:

- Cuando la distancia entre ellos puede corresponder a un hueco en la pared.

- Cuando se detecta cambio de lado, es decir, que pertenece a la otra pared del túnel.
- Cuanto se detectan cambios de signo en ΔX.

. En la Figura 11 se aprecian los 4 segmentos *nseg* encontrados a partir de la nube de puntos (en rojo). La información de los segmentos corresponde al número del punto final del segmento (*fins*).

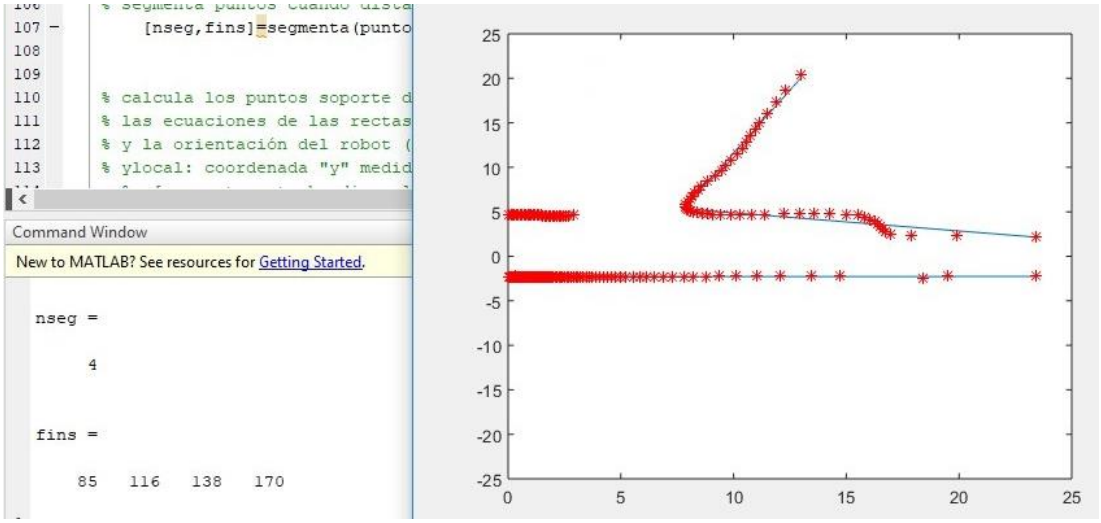


Figura 11. Segmentación de los puntos de un barrido del láser.

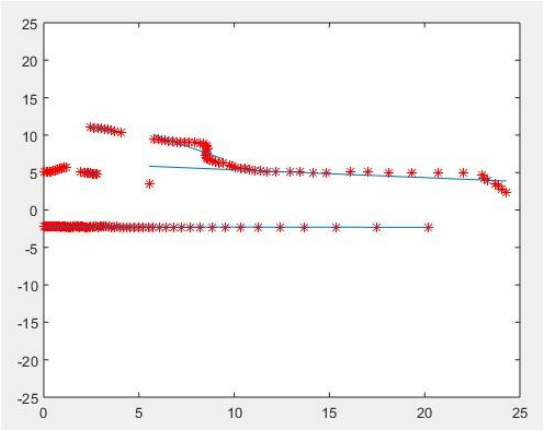


Figura 12. Galería 10 del túnel de Somport.

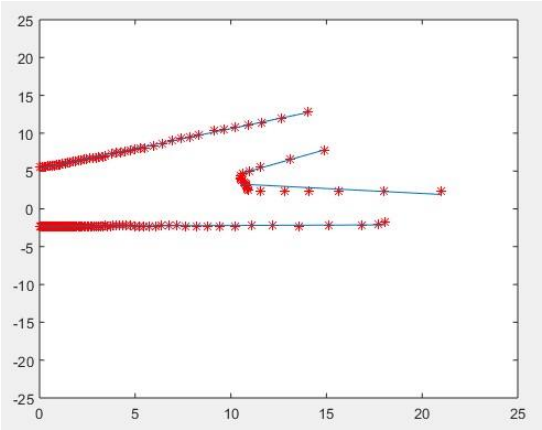


Figura 13. Galería 5 del túnel de Somport.

En las Figuras 12 y 13 se muestran dos de las galerías del túnel de Somport donde se puede ver la segmentación de los puntos que soportan las rectas dibujadas en azul. El cálculo de las rectas se describe en el punto siguiente. El resto de galerías y su segmentación se encuentran en el Anexo C. Obsérvese que las galerías son muy diferentes entre sí, por lo que una segmentación general correcta es esencial para el método.

4. Cálculo de las rectas que forma cada segmento:

Si el número de puntos que va a tener la recta supera el mínimo exigido como soporte de recta (en azul en Figura 14), se calcula su ecuación. Nótese que las rectas calculadas no corresponden siempre con puntos alineados, puede corresponder a un chaflán. No se refina más, ya que no es necesario para el método propuesto, reduciendo

el coste computacional y la proliferación de segmentos innecesarios. Además, precisamente, esto permite detectar cuando finaliza la galería y se regresa a un tramo de paredes paralelas en el túnel, siendo ese el momento de la localización de la galería.

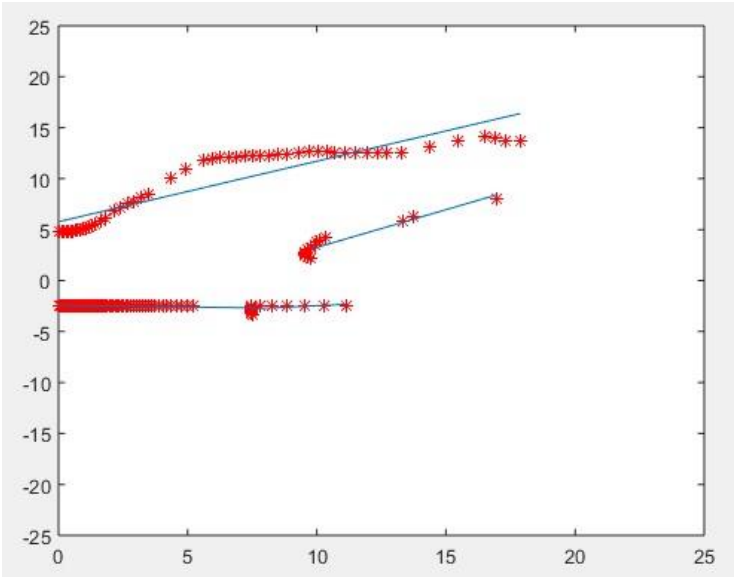


Figura 14. Cálculo de rectas y segmentos en un barrido del láser en la galería 9.

.

5. Realineamiento de puntos:

Una vez tratados los puntos y formado las rectas con ellas, se reorientan los puntos para alinearlos con el eje X de la referencia absoluta.

6. Identificación de galerías:

La condición para determinar si puede existir una galería es que la distancia entre ambas paredes sea mayor que la del parámetro ajustable de discontinuidad mínima de galería. Si cumple esta condición, tras verificar si es perteneciente a una galería a izquierda o a derecha, se verifica que el segmento que ve frontalmente el escáner (el segundo si es pared izquierda, primero si es derecha, pared 3 y 4 en Figura 5, respectivamente) contiene el mínimo de puntos soporte definido para ello, y si su pendiente es mayor que el parámetro ajustable de mínima pendiente de inclinación que pueda tener la galería.

7. Validación de galería:

Para validar una galería es necesario verificar que se detecta un número de veces consecutivo definido por un parámetro de persistencia ajustable. Así se evita que alguna detección espúrea debida a las medidas pueda conducir a un resultado erróneo.

Una vez validada la galería, su localización se utilizará para actualizar la coordenada absoluta X del robot, ya que se conoce con una buena precisión la localización de la galería en el mapa topológico. En el Capítulo siguiente se explica cómo se calcula esta posición.

3. Localización absoluta del robot en un túnel

3.1. Cálculo de la posición del robot

Como se ha explicado en el Capítulo anterior, la actualización de las 3 componentes de la localización se realiza utilizando información diferente. Las coordenadas *Y* y θ del robot se actualizan en cada iteración a partir de la información de rango que permite obtener la situación relativa de las paredes y el robot. La coordenada *X* se actualiza en cada periodo con la información odométrica, que acumula error, salvo cuando se reconoce una galería, que permite reducir el error de localización al error de localización de la galería disponible en el mapa topológico.

- Cálculo de *Y* y θ:

Este cálculo se realiza en cada iteración una vez construidas las rectas que forman las paredes a partir de la nube de puntos láser. Lo primero que se busca es que la toma de referencia sea el centro del túnel y para ello se busca que ambas paredes del túnel sean paralelas. Así se guarda la referencia a la pared izquierda y a la pared derecha para necesidades posteriores. Una vez encontradas estas referencias se distinguen dos casos:

- Que las paredes del túnel sean paralelas.
- Que las paredes del túnel no sean paralelas.

- Paralelismo:

En caso de que haya paralelismo entre las paredes, se calcula la orientación θ (heading) como:

$$heading = -\ atan(m)$$

Siendo *m* la pendiente de cualquier pared.

También se calcula la distancia que hay entre la referencia del láser y las rectas obteniendo así la distancia a ambas paredes. Calculando la diferencia entre ambas distancias, obtenemos la *ylocal* (‘d’ en las Figuras 15, 16, 17 y 18), que no corresponde con la *Y* absoluta.

$$ylocal = ydcha - yizda$$

Se presentan así, los 4 casos correspondientes a la situación relativa del robot respecto al eje del túnel y a la orientación absoluta, como se aprecian en las Figuras 15, 16, 17 y 18. Los valores calculados para *Y* absoluta se representan en cada una de ellas.

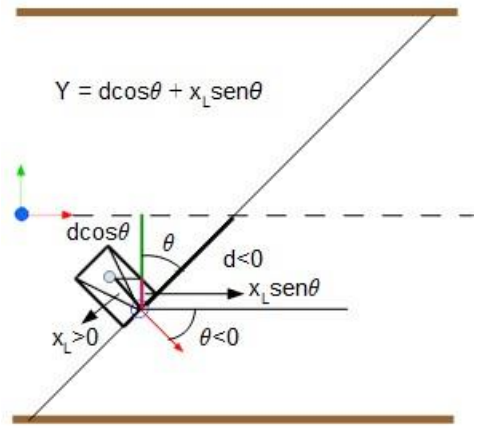


Figura 15. Caso 1: Ylocal < 0, θ < 0, xL > 0.

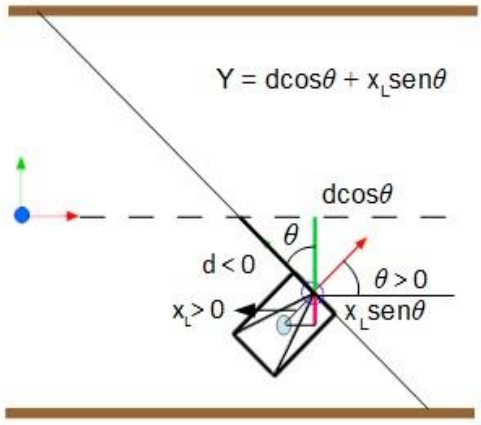


Figura 16. Caso 2: Ylocal < 0, θ > 0, xL > 0.

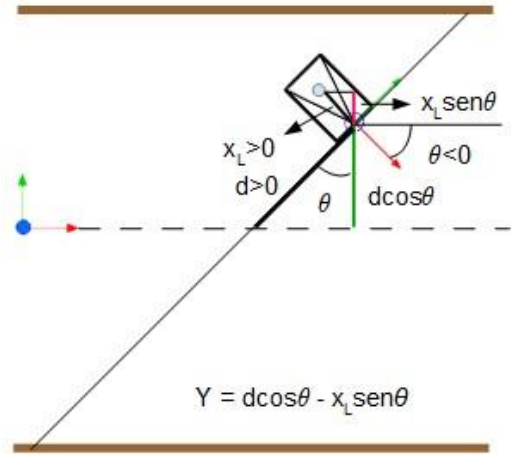


Figura 17. Caso 3: Ylocal > 0, θ < 0, xL > 0.

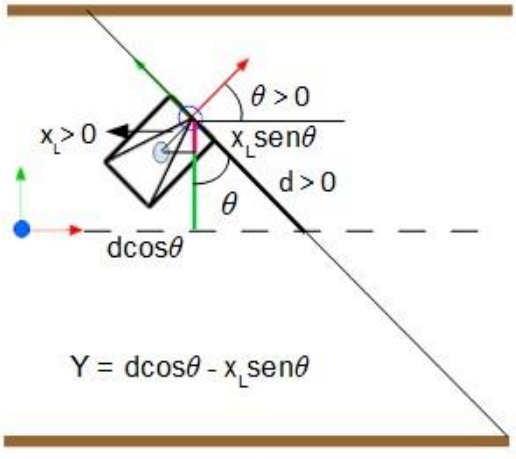


Figura 18. Caso 4: Ylocal > 0, θ > 0, xL > 0.

En el caso de ensanchamiento del túnel, la coordenada *Y* se calcula con respecto a la pared que no ha experimentado el ensanchamiento. Este cálculo se explicará en el apartado de no paralelismo ya que es el mismo a diferencia de la condición usada para discriminar sobre qué pared calcular la *Y*.

Finalmente en caso de que aun con esta condición se superase el parámetro de ensanchamiento definido, se mantendrían la *ylocal* y la θ del paso anterior ya que se toma como hipótesis que el robot intentará viajar por el centro del túnel.

- No paralelismo:

Cuando se produce ensanchamiento, habitualmente la pared que provoca el ensanchamiento, o la recta que contiene los puntos soporte, cambian de orientación, eligiendo como pared de referencia para el cálculo aquella que experimenta menos o nula variación de orientación. Como esta diferencia realizada a cada iteración podría suponer saltos entre rectas que falseasen la medida, esta diferencia se calcula cada cinco iteraciones.

Una vez se comprueba que pared ha variado menos se procede a calcular la θ igual que en los casos de paralelismo, con la pendiente de la recta de la pared izquierda

en caso de tomar esta, o la de la derecha en el caso contrario. La *ylocal* varía en función de qué pared estemos utilizando obteniendo:

$$ylocal = y_o izda - yizda$$
$$ylocal = ydcha - y_o dcha$$

El signo de *ylocal* indica a qué pared se ha acercado o alejado. El cálculo de la coordenada *Y* con la pared izquierda se obtiene:

$$Y = y_o izda - y_L cos\theta - x_L sen\theta$$

Para la pared derecha:

$$Y = y_R cos\theta - x_L sen\theta - y_o dcha$$

Finalmente en caso de que aun con esta condición se superase ese parámetro de ensanchamiento, o bien que a pesar de que calculando a cada cierto número de iteraciones se produzcan saltos entre la *Y* absoluta anterior y actual o entre θ absoluta anterior y actual, más grandes que un parámetro ajustable de 'salto', se mantendrían la *ylocal* y la θ del paso anterior ya que se toma como hipótesis que el robot intentará viajar por el centro del túnel y de una iteración a otra el robot no puede desplazarse ni girarse bruscamente.

- Cálculo de *X*:

El cálculo de la *X* se ha obtenido mediante la intersección formada por, según el sentido del barrido, el segundo segmento de la galería si está a la derecha (Recta clave G2 en Figura 19), y el primero de la galería si está a la izquierda (Recta clave G1 en Figura 19), y la pared paralela que sigue a la galería, P1 en el caso de de Recta clave G1 y P2 en el caso de Recta clave G2.

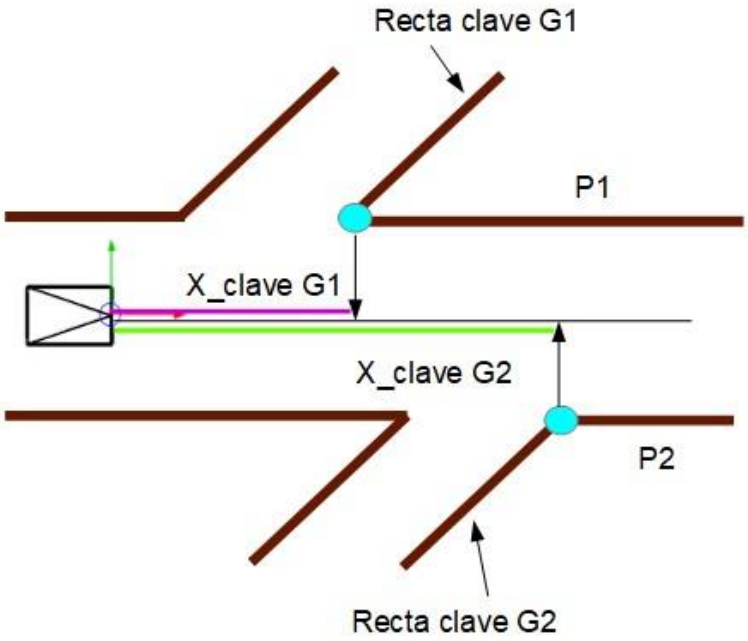


Figura 19. Cálculo de intersección.

Estas rectas ahora denominadas como recta clave, son las formadas por más puntos, entonces, cuando se empieza a detectar una galería, en cada posición del robot, se van guardando esas rectas y comparando cuál de ellas tiene mayor número de puntos hasta quedarnos con ella.

Una vez obtenida esta recta y teniendo validada la galería, se procede a buscar la recta P1, o P2 en caso de la derecha, (Figura 19), una recta que sea larga y paralela a la pared de enfrente. Después con las dos rectas, se calcula la intersección de ellas obteniendo así el punto de corte. Es la posición relativa de la galería vista por el robot, y corresponderá con el punto de referencia de la galería, contenido en el mapa topológico. Se utiliza para corregir la localización absoluta del robot, cada vez que detecta una galería.

3.2. Corrección de la localización

La corrección de localización se realiza de dos formas, una para corregir la *Y* y la θ que se realiza en cada iteración y otra para corregir la *X* cuando encuentra una galería.

Durante cada iteración, conocida la localización absoluta anterior, la localización relativa en *X* entre el paso anterior y el actual, la *Y* absoluta actual y la θ absoluta actual, calculamos la localización absoluta actual mediante matrices de transformación.

$$
\begin{aligned}
{}^{W}x_{R1} &= [X_{ant}, Y_{ant}, \theta_{ant}] \\
{}^{R1}x_{R2} &= [X_{rel}, 0.0, 0.0] \\
{}^{W}T_{R2_x} &= \hom({}^{W}x_{R1}) * \hom({}^{R1}x_{R2}) \\
{}^{W}x_{R2_x} &= loc({}^{W}T_{R2_x}) \\
{}^{W}x_{R2} &= [X_{{}^{w}x_{R2_x}}, Y_{act}, \theta_{act}]
\end{aligned}
$$

Siendo R1 y R2 las referencias en un instante de muestreo y el siguiente, y W la referencia absoluta (*World*). Del vector de localización ${}^{W}X_{R2x}$ se obtiene la coordenada *X* absoluta actual para formar el vector de localización actual, ${}^{W}X_{R2}$, con las coordenadas *Y* y θ absolutas calculadas como se ha explicado en la sección 3.1.

Por otro lado, cuando se encuentra y valida una galería, hay que tener en cuenta en qué iteración se ha obtenido la recta correspondiente a la pared lateral de la galería frontal al robot, para realizar la corrección delocalización una vez calculado el punto de galería (G), explicado anteriormente. Como se puede ver en la Figura 21, la intersección es calculada cuando se encuentra P1 y corresponde con la situación del robot en la Figura 20. Por tanto, hay que relocalizar en todas las posiciones desde que se encuentra la recta clave (posición del robot en Figura 20), hasta la posición actual (posición del robot en Figura 21).

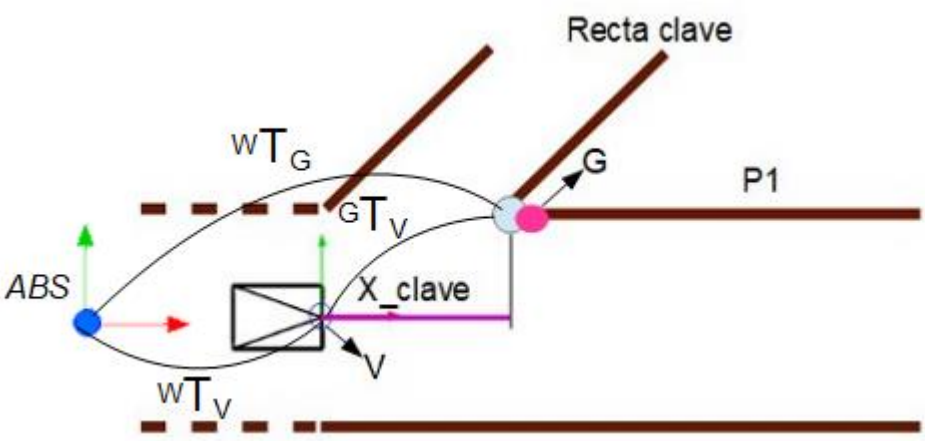


Figura 20. Instante en el que se guarda la recta clave definitiva para el cálculo de la intersección.

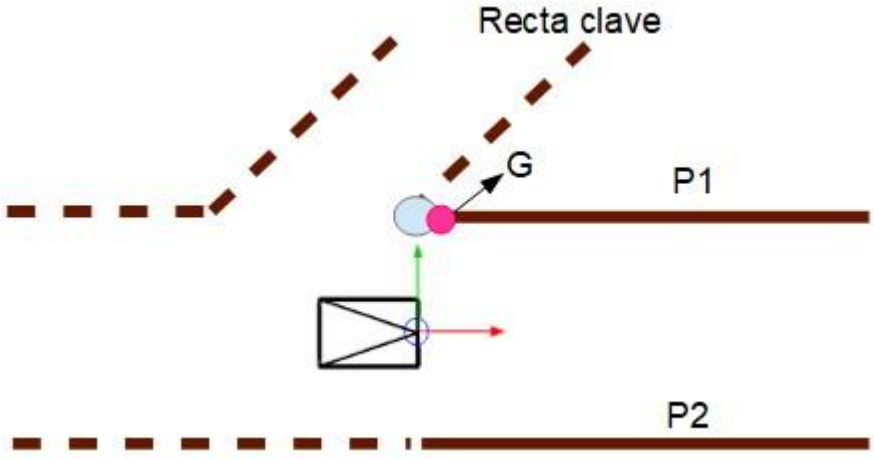


Figura 21. Instante en el que se realiza el cálculo de la intersección.

El punto de intersección corresponderá al punto de referencia de la coordenada X de la galería en el mapa topológico, referencia G (ver Figura 20). Por tanto es necesario corregir la coordenada *X* desde que se encontró la recta clave, referencia V (ver Figura 20), utilizando para ello las transformaciones relativas consecutivas entre las referencias.

Los cálculos a realizar son:

$$^{W}x_{G} = [X_g, 0.0, 0.0]$$

$$^{G}x_{V} = [X_{clave}, 0.0, 0.0]$$

$$^{W}T_{V} = \hom(^{W}x_{G}) * \hom(^{G}x_{V})$$

$$^{W}x_{V} = loc(^{W}T_{V})$$

Donde $^{W}X_{G}$ es el vector de localización del punto de referencia de la coordenada X de la galería en el mapa topológico, $^{G}X_{V}$ corresponde al vector de localización de la posición del robot en la Figura 20, momento en el que encuentra la recta clave. Todas estas transformaciones se pueden ver en la Figura 20.

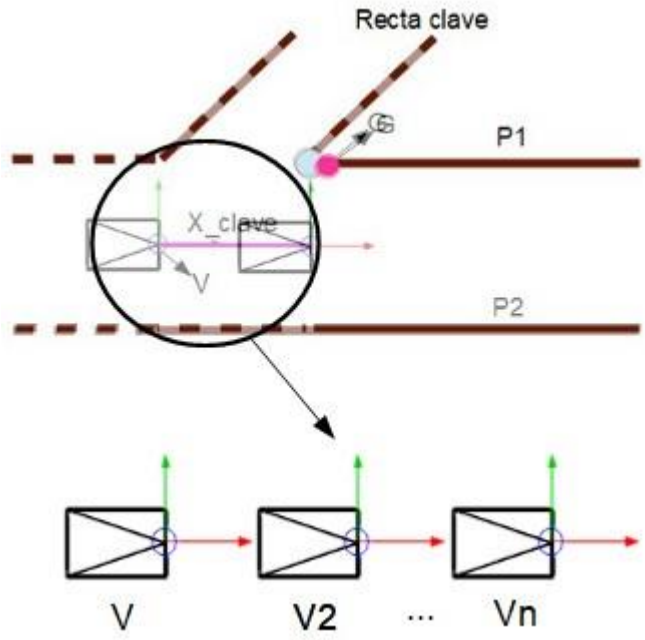


Figura 22. Relocalización.

Una vez corregida la localización de ese punto, se procede a corregir las posiciones siguientes hasta llegar a la posición actual, V_n (ver Figura 22), con la posición relativa en *X*, manteniendo la *Y* y la θ absolutas ya que ya han sido actualizadas en su momento:

$$
\begin{aligned}
{}^{W}x_{V_{n-1}} &= [X_{V_{n-1}}, 0.0, 0.0] \\
{}^{V_n^{-1}}x_{V_n} &= [X_{rel}, 0.0, 0.0] \\
{}^{W}T_{V_n\,x} &= \hom\left({}^{W}x_{V_{n-1}}\right) * \hom\left({}^{V_n^{-1}}x_{V_n}\right) \\
{}^{W}x_{V_n\,x} &= loc\left({}^{W}T_{V_n\,x}\right) \\
{}^{W}x_{V_n} &= [X_{{}^{w}x_{V_n\,x}}, Y_{V_n}, \theta_{V_n}]
\end{aligned}
$$

Donde ${}^{W}X_{Vn-1}$ representa el vector de la localización absoluta del instante anterior, ${}^{Vn-1}X_{Vn}$ es el vector de localización relativa entre instante anterior y actual y ${}^{W}X_{Vn}$ es el vector de la localización absoluta del instante actual, posición alcanzada en la Figura 21 que es cuando se valida la galería.

4. Resultados obtenidos con los datos del túnel de Somport

Una vez simulado el algoritmo con los datos proporcionados del túnel de Somport obtenidos con el Galloper se obtiene lo siguiente:

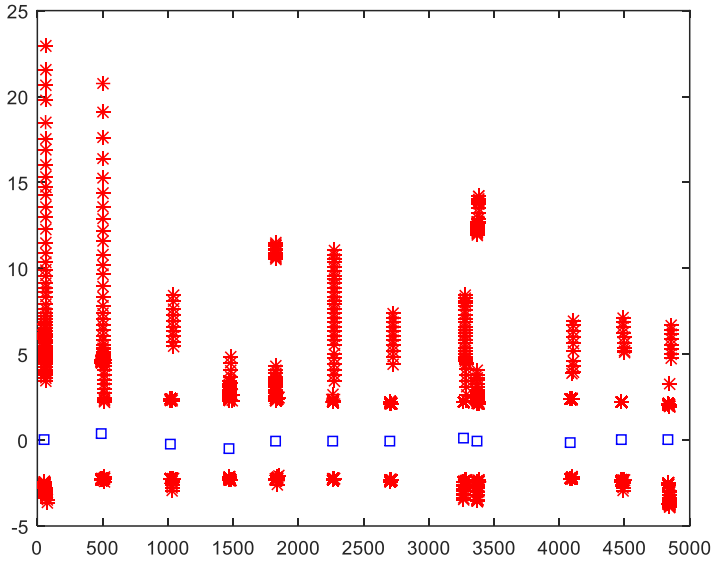


Figura 23. Galerías encontradas.

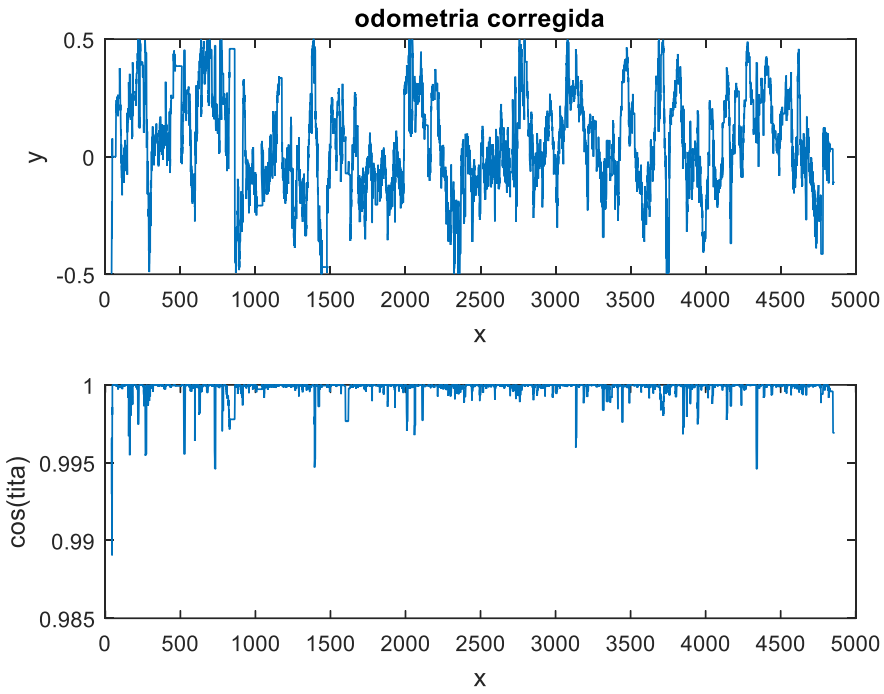


Figura 24. Odometría corregida en base a galerías y paredes del túnel.

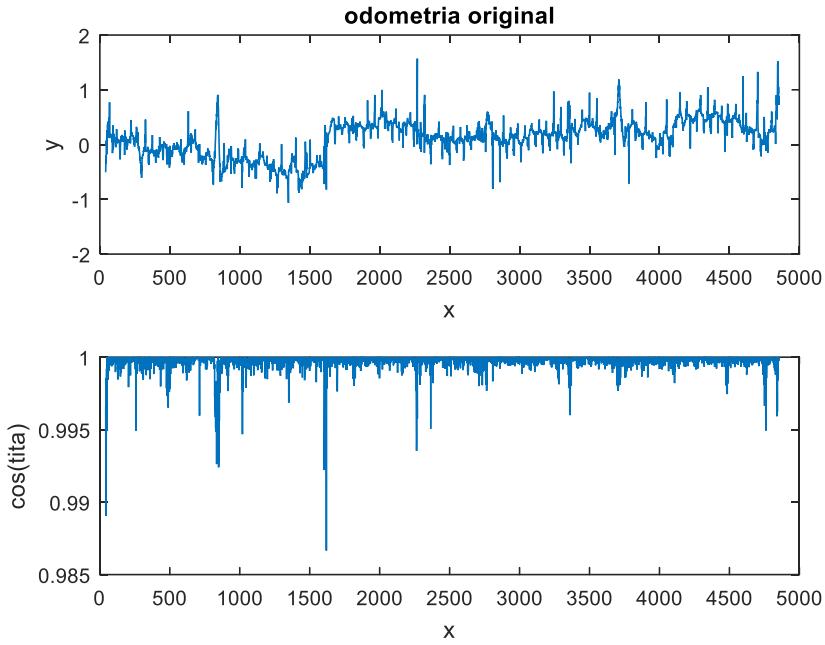


Figura 25. Odometría original obtenida por el robot.

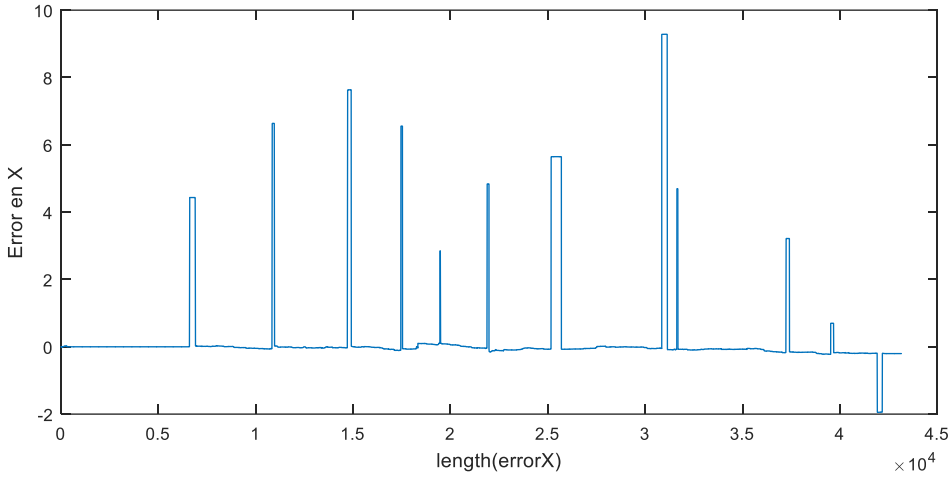


Figura 26. Error en X.

```
g =

[  1.0, 51.862,         0,           0,  6614.0, 47.433,   0.024471,  0.082338]
[  2.0, 492.55,   0.38527,   -0.011632, 10837.0, 485.97,   -0.27658,  0.038034]
[  3.0, 1020.3,  -0.20791,     0.02355, 14718.0, 1012.7,   -0.33485,  0.012268]
[  4.0, 1463.1,  -0.46938,   0.0096706, 17458.0, 1456.7,   -0.62842,  0.029176]
[  5.0, 1824.5, -0.088901,   0.0031373, 19466.0, 1821.6,     0.2635,  0.040209]
[  6.0, 2260.7, -0.091583,  -0.0025971, 21878.0, 2255.9,    0.33102,  0.026555]
[  7.0, 2703.9, -0.067096,    0.018407, 25171.0, 2698.3,    0.15671,  0.038564]
[  8.0, 3263.8,   0.14018,  -0.0068944, 30851.0, 3254.5,    0.21316, -0.013408]
[  9.0, 3370.5,  -0.07912,    0.039675, 31628.0, 3365.9,   -0.032342, -0.013498]
[ 10.0, 4086.7,  -0.12805, -0.00019906, 37237.0, 4083.7, 0.00015129,  0.028383]
[ 11.0, 4473.0,  0.023479,   0.0028656, 39526.0, 4472.6,    0.33998,   0.01727]
[ 12.0, 4834.5,   0.03368,    0.028796, 41919.0, 4836.7,    0.73373,  0.045915]
```

Figura 27. Las 12 galerías encontradas: por columnas número de galería, Xcorregida, Ycorregida, θ corregida, paso de muestreo en el que se ha validado, Xodometría, Yodometría, θ odometría.

En la Figura 23 se representan las 12 galerías encontradas. El 100% de las existentes en el tramo de túnel de aproximadamente 4800 metros (en rojo) y las posiciones del robot en el momento de la actualización de la localización en cada galería (en azul).

En la Figura 24 y 25 se muestran las odometrías corregida y original respectivamente del robot. Como se puede observar se ve una clara mejora, alrededor del 1.5m en media, de la odometría respecto de la coordenada *Y*. El túnel de Somport no contempla giros ni intersecciones con lo cual el robot debería seguir el eje longitudinal sin muchas variaciones respecto de la referencia absoluta inicial. Por otro lado, en la Figura 26 se muestra los errores en X cuando se integran las medidas de odometría y las de las galerías reconocidas. El error es nulo entre galerías ya que en todos esos tramos la localización se obtiene directamente de la odometría, al no haber información adicional. En las posiciones en las que se reconoce una galería, aparece un error, que es precisamente el error acumulado que corrige el método propuesto. Como a partir de la localización de cada galería, esta odometría es la que se utiliza como nueva localización el robot, el error vuelve a ser nulo hasta la siguiente galería. Esto quiere decir que, en caso de no hacer las correcciones de localización, el error iría acumulándose, hasta alcanzar en este caso alrededor de 58 metros al final del trayecto (suma de errores en la Figura 26). Los errores positivos indican que el robot estaba más adelantado respecto de donde indicaba la odometría y mientras que el negativo de la última galería indica que estaba más retrasado respecto de donde indicaba. En consecuencia, comparando Figuras 24 y 25, la odometría corregida tiene como máximo el error que pueda proporcionar el sensor láser escáner frontal al localizar la galería, y el debido al error de localización del punto de referencia de la galería en el mapa topológico. Mientras que la odometría original conduce a un incremento continuo del error de localización. Cabe mencionar que en el caso del experimento presentado, el error de la odometría pura sería mucho mayor. En realidad la odometría utilizada está corregida mediante una técnica de scan matching (denominada también *laser odometry*). Pero, a pesar de ello, se puede comprobar que existe un error acumulativo, si no se utilizan técnicas de corrección como la presentada en este trabajo.

En la Figura 27 se muestran las doce galerías encontradas del túnel de Somport. Las coordenadas Y y θ de la primera galería tienen ese valor porque la simulación comienza con la galería y no dispone de paredes paralelas para calcular esos valores.

También se han simulado los datos de vuelta del túnel, obteniendo los siguientes resultados:

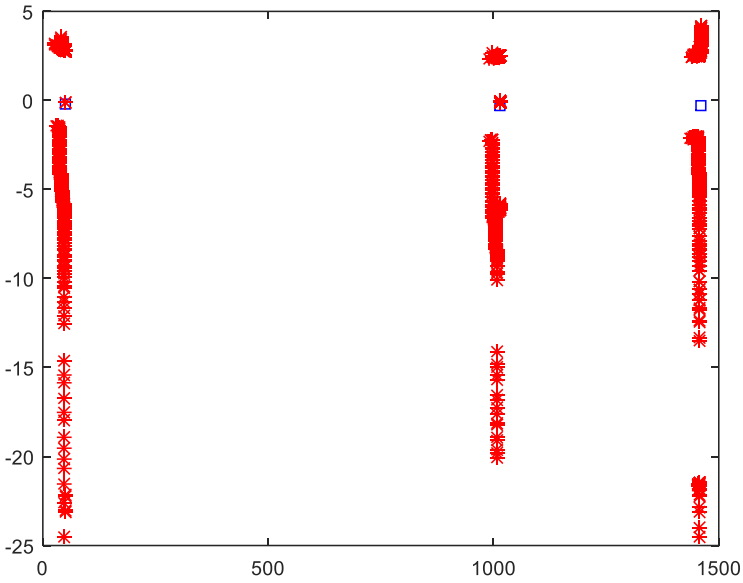


Figura 28. Galerías encontradas sentido de vuelta.

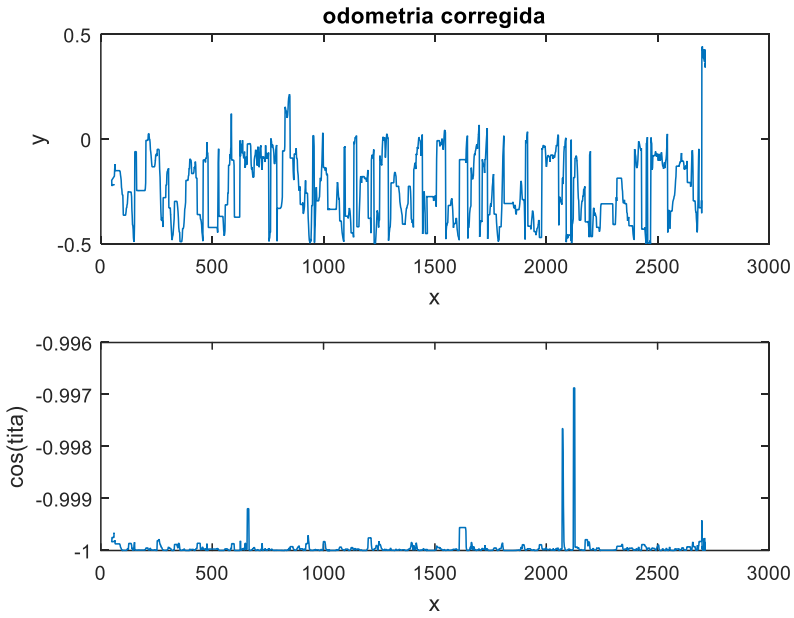


Figura 29. Odometría corregida sentido de vuelta.

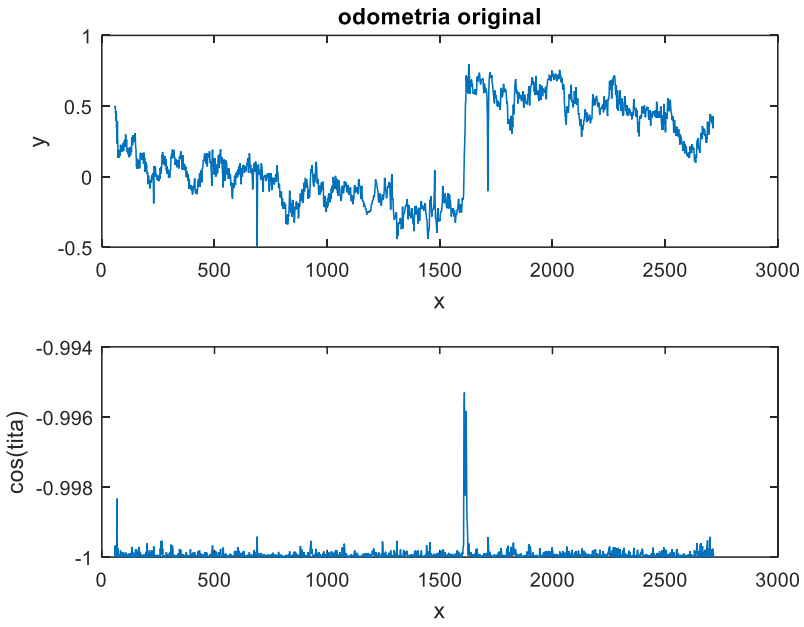


Figura 30. Odometría original obtenida por el robot sentido de vuelta.

```
g =

[ 4.0, 1443.6, -0.28247, -3.1394, 753.0, 1487.3,  -0.39923,  3.1268]
[ 3.0, 1012.6, -0.33498, -3.1399, 993.0, 1038.6, -0.085162, -3.1373]
```

Figura 31. Galerías sentido de vuelta: por columnas número de galería, Xcorregida, Ycorregida, θ corregida, paso de muestreo en el que se ha validado, Xodometría, Yodometría, θ odometría.

Un factor importante para el reconocimiento de galerías es la velocidad del robot a la que se adquiere la información de rango. Si no hay suficientes puntos soporte, el reconocimiento no es correcto. Este problema se pone de manifiesto con los datos tomados en el trayecto de vuelta, a una velocidad superior que en el de ida, a más de 3 m/s en media. Como se puede apreciar en la Figura 28, el robot comienza el sentido de vuelta en aproximadamente los 2750 metros y termina antes de llegar a la primera galería, lo que implica que debería localizar 5 galerías, habiendo reconocido solo 2. Este experimento permite acotar la velocidad de captura para tener un buen reconocimiento, aproximadamente de 1 m/s.

5. Implementación en ROS

Robot Operating System, también conocido como ROS [9], es un paquete software estándar ampliamente utilizado en el campo de la robótica. Se ha utilizado conjuntamente con Gazebo, herramienta software para simulación. En ambos paquetes hay una amplia variedad de modelos de robots, sensores, escenarios, y algoritmos de navegación y localización, para simular de una manera realista una aplicación. El desarrollo realizado preliminarmente en MATLAB para la puesta a punto y validación inicial del método propuesto, se ha trasladado al entorno ROS-Gazebo para una simulación y ajuste final realista. A diferencia de los realizado en MATLAB en el capítulo anterior, donde se ha trabajado con datos reales capturados previamente, y reproducida *offline* la localización sobre el mapa láser, el entorno ROS-Gazebo permite simular diferentes escenarios, a la vez que se puede visualizar la tarea y todas las variables de interés utilizadas.

ROS trabaja con lo que se denominan *nodos* y *topics*, siendo los *nodos* los programas ejecutables y los *topics* las 'tuberías' que envían la información. Funcionan de tal forma que los *nodos* se suscriben a los *topics* para recibir información y a la vez estos *nodos* publican *topics* para enviar información. Todo este software se puede juntar en paquetes que constituyan un módulo.

El algoritmo explicado en los capítulos anteriores se ha implementado en ROS, y se han utilizado otros paquetes existentes en él para una localización más realista del robot. Existen paquetes de localización que permiten la integración de distintas medidas de sensores para mejorar dicha localización, por ejemplo sensores de odometría, inerciales IMU, cámaras, y escáneres láser entre otros. La integración se realiza mediante un filtro Extendido de Kalman (EKF), que permite integrar diferentes informaciones con su incertidumbre, para obtener finalmente un valor de la variable estimada con una incertidumbre más reducida, aumentando por tanto la precisión. En este trabajo se ha utilizado para la estimación precisa de la localización del robot (x,y, θ), integrando la información odométrica con incertidumbre obtenida del simulador, con la información de localización del robot obtenida a partir de la información de los escáner láser (LIDAR) con el método propuesto en el capítulo anterior.

En la Figura 32 se puede ver un pequeño esquema de como quedaría implementado en este proyecto en particular. Del algoritmo principal, nodo */scan*, se obtiene la medida odométrica del simulador, */odom*, y la localización del robot obtenida a partir de la información del escáner láser, */galleries*, cada una con su incertidumbre. Integradas ambas en un filtro de Kalman, se obtiene la estimación de odometría, */odom_estimada*, también con su incertidumbre. En el caso de la */odom*, la media es la medida que se toma de cada ciclo. Para el cálculo de covarianza se han estimado los errores que puede haber en cada medida. Respecto del valor de *X* se ha estimado que el error es de 2 cm por metro avanzado, el cual, va incrementando a cada ciclo al no reconocer galerías. Respecto al valor de *Y*, vistas las simulaciones del Capítulo siguiente, se ha tomado que es de 1 cm por metro avanzado y respecto al valor de θ, de 0.005 radianes por metro avanzado. Ambos constantes ya que se conocen sus medidas a cada ciclo. En el caso de */galleries*, la media resulta de 0 hasta que reconoce una galería. El error estimado cuando no reconoce galería es de un valor muy alto para que el filtro de Kalman ignore esa medida. Una vez encontrada, el error se estima en 1.5 m debido al error que produzca la medida tomada del mapa topológico. Por otro lado, la odometría estimada, */odom_estimada*, sus valores son calculados por el filtro siendo sus salidas, la

media ponderada de ambas entradas. Esta odometría estimada se debería realimentar en cada ciclo de ese valor ya que es el mejor que puede obtener.

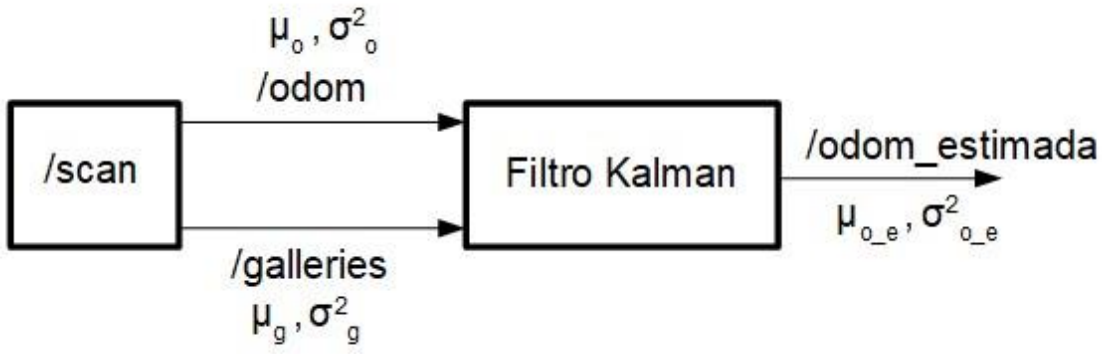


Figura 32. Esquema de la implementación de un filtro de Kalman.

Se ha intentado implementar dos de los paquetes que se proporcionan en ROS, *robot_localization* y *robot_pose_ekf*. Tras implementar ambos, los resultados obtenidos no han sido correctos, por tanto se deja como trabajo futuro.

ROS también cuenta con un visualizador muy completo, RVIZ, donde se muestra al robot y se pueden ver la nube de puntos láser que está capturando el robot en una posición determinada y el *goal* o meta a donde tiene que llegar al robot.

Como lenguaje de implementación del algoritmo localizador de galerías se ha utilizado Python. A diferencia de trabajar con MATLAB, los datos se tratan de manera *online*. Simula la información sensorial capturada en tiempo real durante la navegación simulada, calculándose la localización en tiempo real durante la navegación. La información es enviada por el *topic*, ejecutando el *nodo* correspondiente al algoritmo de localización.

En la Figura 33 se puede ver el nodo principal donde está implementado el algoritmo de este proyecto, /scan, que está suscrito al topic del laser frontal, /laser_front/scan, y a la localización del simulador, /loc, que ha tenido que ser transformada por el nodo /odomloc, para cambiar el tipo de mensaje recibido por /sim_p3at/odom.

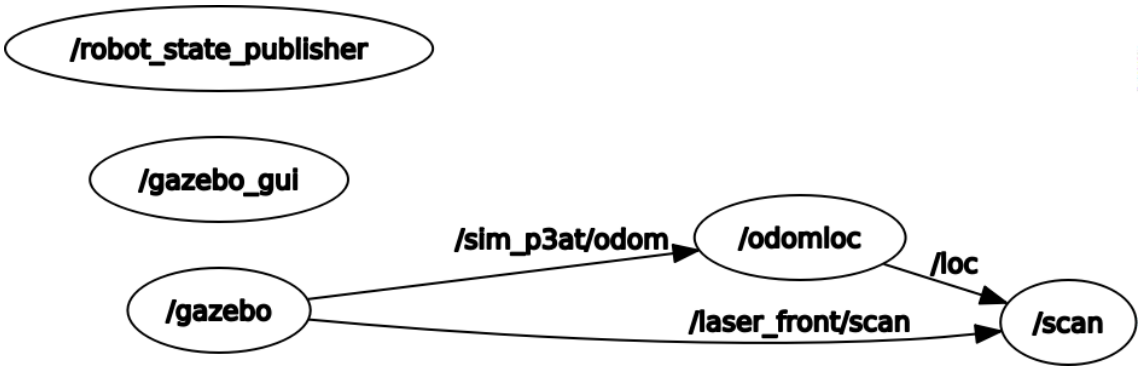


Figura 33. Nodos y topics principales.

Todos los algoritmos creados en ROS se encuentran en el apartado de Anexos.

6. Creación de escenarios en Gazebo

Con el simulador Gazebo se pueden crear escenarios con distintas características para probar el algoritmo. Se basa en la creación de modelos que luego se añaden a un fichero para la creación del mapa.

Para la creación de modelos en este trabajo, se ha procedido a guardar la pieza de un cubo como la de la Figura 34:

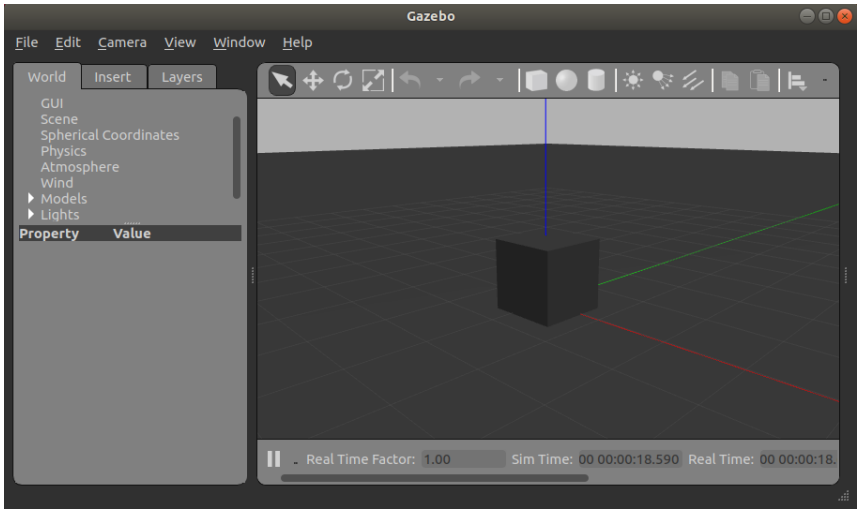


Figura 34. Pieza base cubo en Gazebo.

Para construir el modelo túnel similar al de Somport visto en capítulos anteriores, pero con galerías a ambos lados del corredor principal. Se ha utilizado una forma básica del Gazebo, el cubo, enlazando varios de ellos en distintas posiciones consecutivas para construir el escenario completo. Al conocer la coordenada X de las galerías en un mapa topológico, se pueden situar en esas coordenadas las galerías. Si no se conocen esas coordenadas a priori, se puede construir el escenario deseado, y utilizar el propio localizador del robot proporcionado por el simulador, para calcular los puntos de referencia de cada galería, guiando el robot manualmente, a modo de herramienta de aprendizaje.

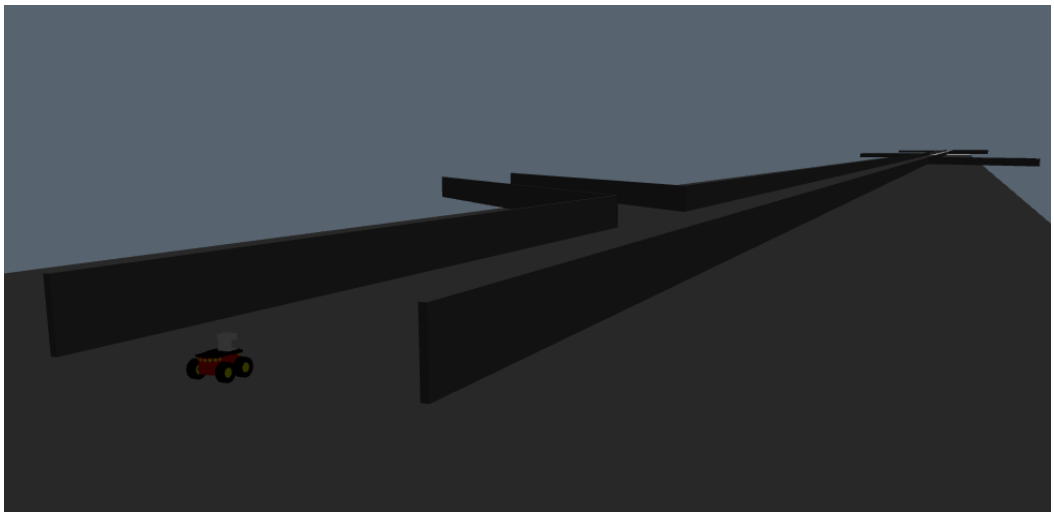


Figura 35. Escenario de trabajo creado.

Como ejemplo principal para este trabajo se muestra el mapa creado en la Figura 36, donde se pueden ver galerías tanto a izquierda como a derecha con distintas orientaciones con inclinaciones de cuarenta y cinco grados y anchuras de galería que varían entre los siete y ocho metros. La anchura del túnel es de cinco metros.

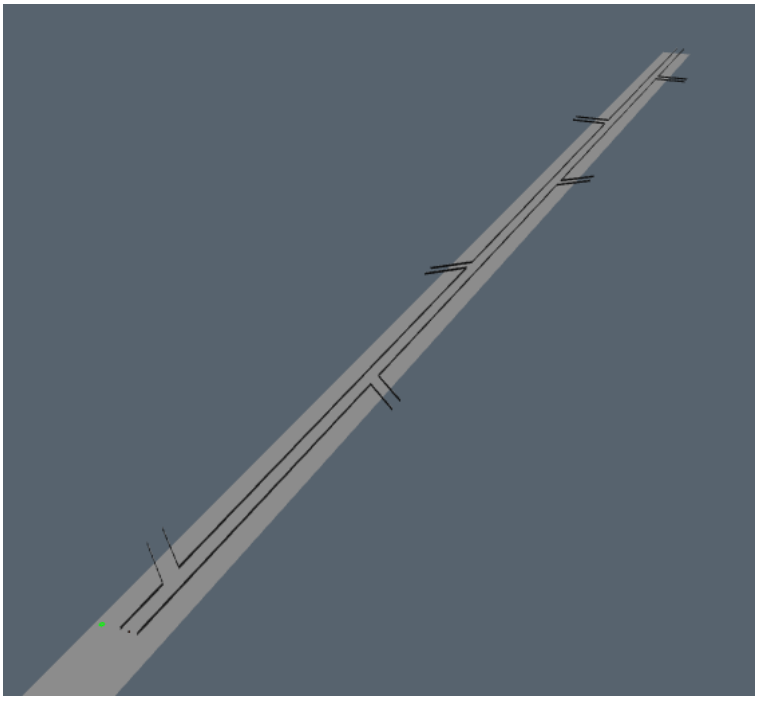


Figura 36. Túnel con 6 galerías.

El modelo de este túnel principal y otros creados para la comprobación del algoritmo se encuentran en Anexo C.

7. Resultados finales

Los resultados que se muestran en este apartado son de la simulación del túnel de la Figura 36. Este túnel contiene los "peores casos" en los que se va a poder encontrar el robot que se irán explicando a lo largo de este apartado, por tanto se determinará el error máximo que se puede cometer con el algoritmo.

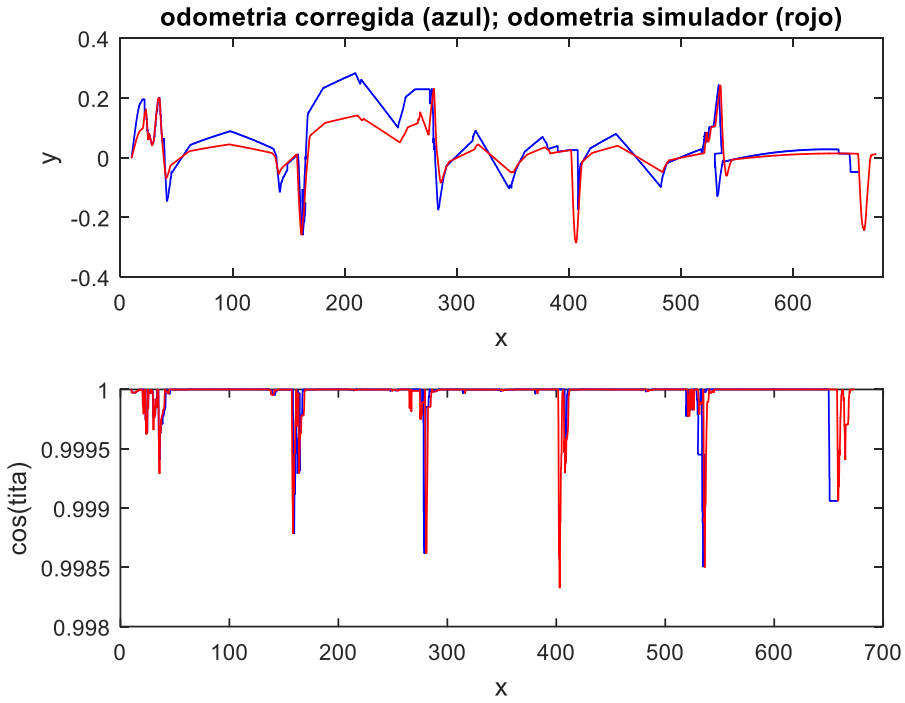


Figura 37. Topic odometría corregida en azul, topic odometría del simulador en rojo.

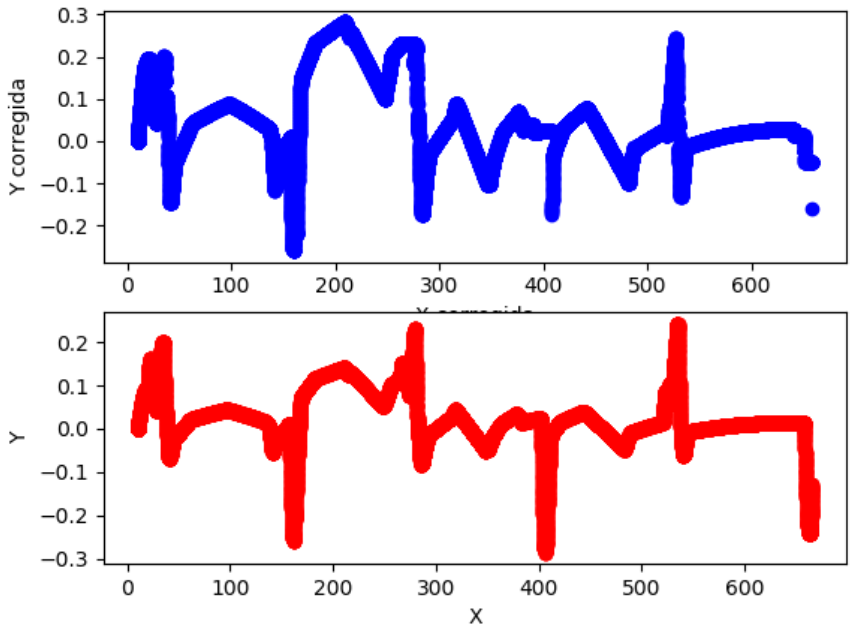


Figura 37. Odometría corregida en azul, odometría simulador en rojo.

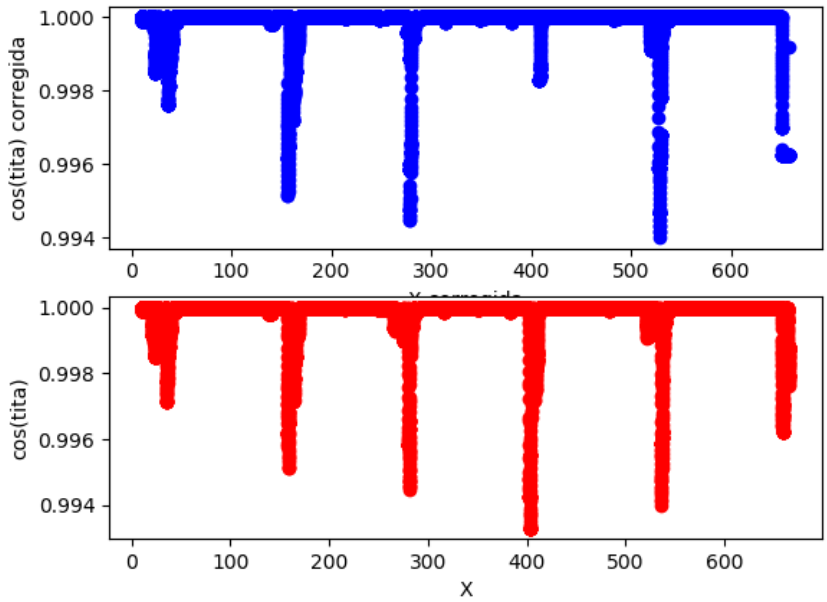


Figura 38. cos(θ) corregida en azul, cos(θ) del simulador en rojo.

Para mostrar los resultados finales obtenidos en simulaciones de Gazebo se ha procedido a guardar en ficheros de texto los mensajes enviados por los *topics* de localización. Al ser la relocalización producida una vez ha pasado la galería, los mensajes antiguos de localización no se pueden modificar como se puede ver en la Figura 37, concretamente en el caso más visual alrededor de X = 540 metros. Por ello, se ha procedido a adjuntar la Figura 38 donde se representa la variable utilizada para guardar la localización, en azul, donde sí se puede ver esta relocalización en X = 540 metros aproximadamente.

Por otro lado, todos los picos producidos en el ángulo en la Figura 37 y 39, son debidos a que el robot lleva un sistema de navegación de seguimiento de paredes. Por tanto cuando se reconoce una galería, el robot sufre desviaciones respecto de su eje longitudinal. El ángulo máximo que se ha producido es de 6.7º en X = 400 como se puede ver en la Figura 39 en la odometría del simulador. El error cometido en ese punto, que es el máximo, es de 0.4º.

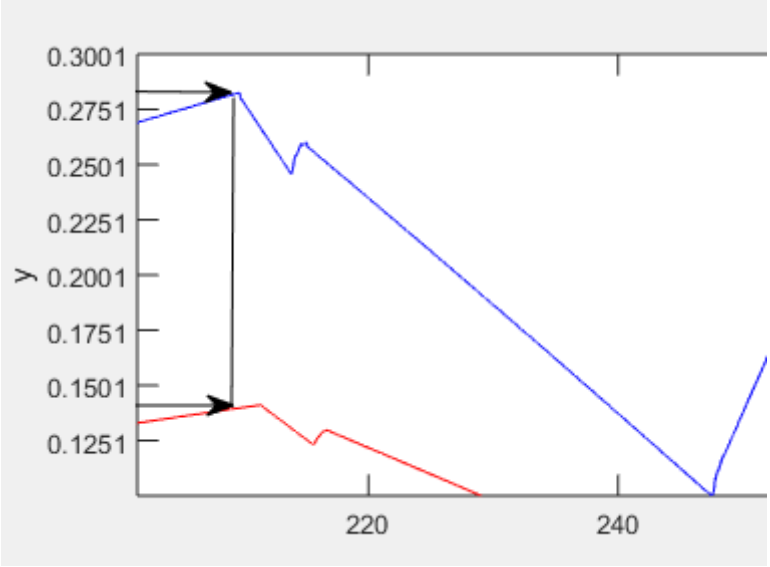


Figura 40. Error máximo en Y.

Como se puede ver en la Figura 37, el error máximo en Y se encuentra alrededor de X = 210 metros, que ampliado en la Figura 40, se puede ver que es de aproximadamente 14 centímetros. Como la odometría ofrecida por el simulador es perfecta, se procede a medir también el error máximo en X que se produce en las galerías 5 y 6 (véase Figura 36). La simulación termina nada más encontrar la galería 6 sin dar un margen, por tanto se procede a hacer un zoom de la galería 5 para medir el error (Figura 41 y 42).

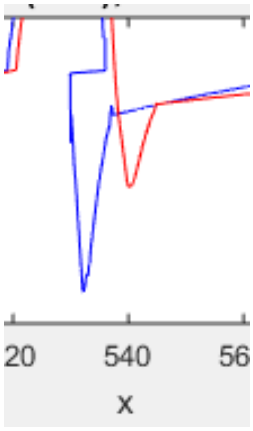


Figura 39. Zoom de la zona que se va a medir en la galería 5.

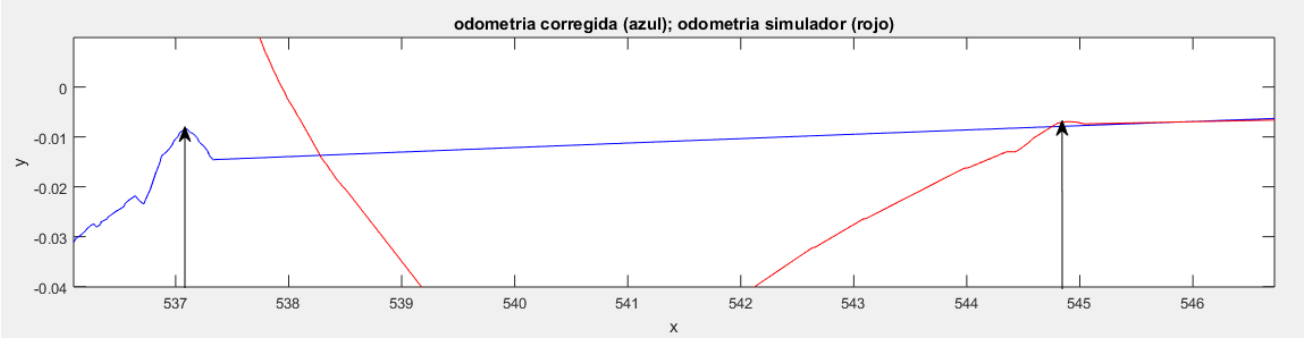


Figura 40. Error medido en X.

En la Figura 42 se puede ver que el error máximo es de 8 metros. Esto es debido a la segmentación producida. Al ser las galerías mucho más estrechas que en el túnel de Somport (véase Anexo C) donde hay anchuras de alrededor de 10 metros debido a los chaflanes, el robot puede ver muchos más puntos alejados como se puede ver en la Figura 43 y 44 que corresponde con las galerías 5 y 6 de la Figura 36. Como se puede intuir la segmentación que hará de una secuencia de puntos que forman un ángulo recto conducirá a un error en la medida, siendo este el peor caso que se puede encontrar.

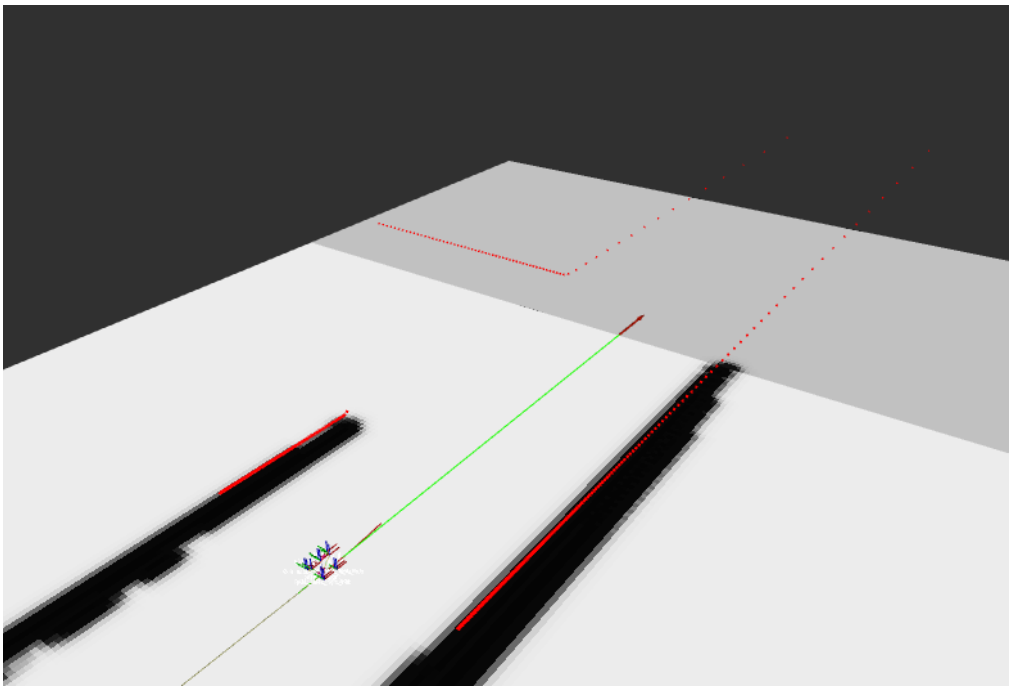


Figura 41. Galería 5.

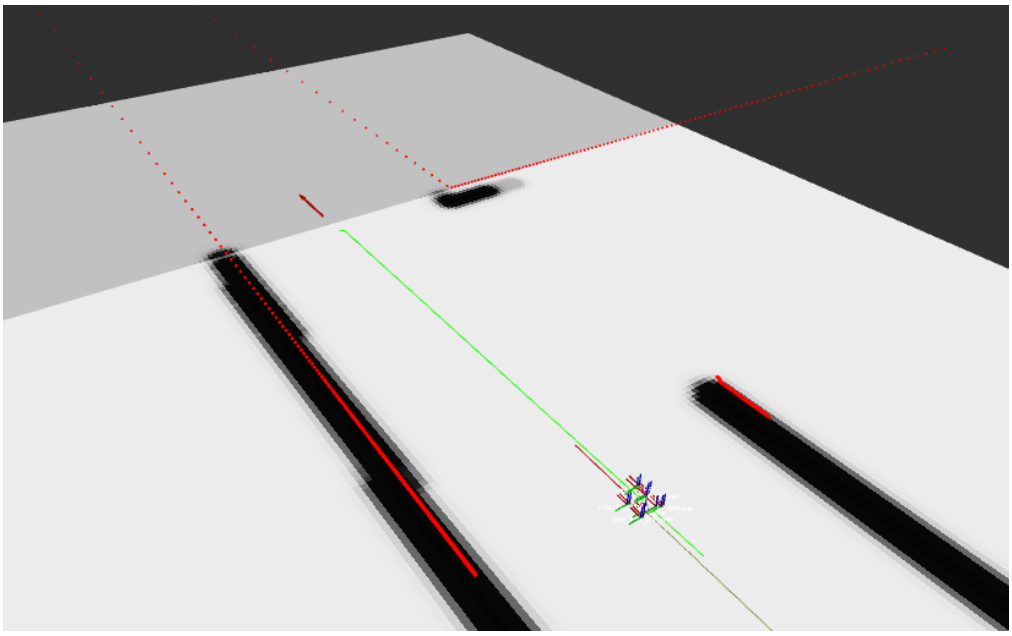


Figura 42. Galería 6.

Como solución queda propuesto elegir la recta que, aparte de exigir un número de puntos soporte, se detecte el momento en el que el robot ve más puntos en la zona de dentro de la galería. Así, cuando empiecen a aumentar los puntos de las paredes

paralelas y disminuyan los puntos que se ven dentro de la galería, se generará menos error de medida.

En el resto de galerías se puede ver que el error en X es muchísimo más pequeño debido al solapamiento que se ve en la Figura 37. Se muestra en la Figura 45 que puede ser aproximadamente de 1 metro, tras la localización de la primera galería.

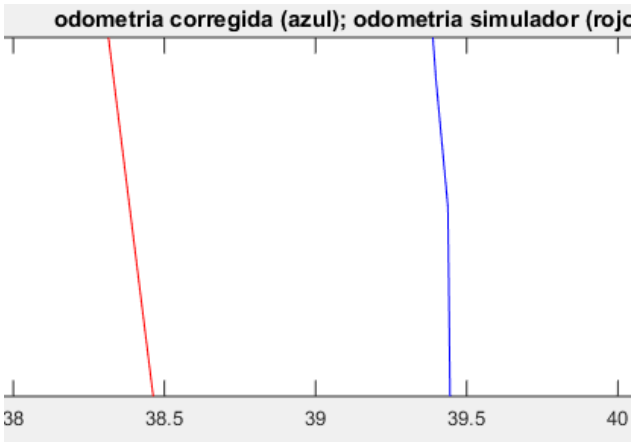


Figura 43. Error en X cerca de la primera galería.

8. Conclusiones

Gracias a las características geométricas reconocibles como pueden ser las galerías, se puede actualizar esporádicamente la localización absoluta de un robot, sin tener un conocimiento previo de la forma o tamaño de las galerías.

Hay factores que por supuesto influyen en la precisión del algoritmo como es la velocidad a la que va el robot ya que a más velocidad, peor reconocimiento de galerías o el sensor láser con el que se van a tomar datos debe tener un rango amplio para que pueda ver tanto el inicio como el fin de la galería para poder reconocerla.

Los pequeños errores que se producen de localización pueden venir tanto de chaflanes que hay en galerías, que no permiten definir nítidamente un punto "esquina" de galería, por lo que hay que elegir un buen punto de *ground truth*, con un cierto margen de error de localización debido a la imprecisión de este punto, como de la desviación de los puntos debidos al error del sensor, que puede ser de 2-5 cm en puntos no muy alejados del sensor. Sin embargo, el error acumulado hasta encontrar la galería se reduce drásticamente hasta valores en el rango del error de localización de la galería en el mapa topológico, que puede ser de 1 a 2 metros. Esto en un escenario de varios kilómetros, es un resultado muy bueno.

A diferencia de otras técnicas que necesitan mapa geométrico previo denso y conocido, como por ejemplo técnicas de "scan matching", la versatilidad obtenida por el algoritmo desarrollado en este trabajo hace que pueda ser utilizado en distintos túneles gracias a los parámetros ajustables y no necesita ser conocido (explorado) previamente. Además el ahorro en coste de tiempo de procesamiento y ocupación de memoria debido a no tener que mantener ni actualizar un mapa geométrico, que es denso, ni la localización en él, es evidente.

La herramienta de Gazebo permite crear modelos de túneles variados aunque tenga pocas piezas para modelar (cubos, cilindros y esferas) y la herramienta rviz para visualizar es muy intuitiva para comprobar si las intersecciones que calcula son buenas.

Como líneas futuras, este proyecto tiene el fin de poderse utilizar en equipos de rescate para diversas funciones como pueden ser el derrumbamiento dentro de un túnel o escapes tóxicos producidos por un camión. En ambos casos se puede utilizar un robot móvil para reconocer el terreno con sensores que midan la presencia de gases, avisen de niveles determinados que puedan ser perjudiciales y poder determinar en qué localización dentro del túnel se encuentra. En función del peligro que haya, los supervisores pueden determinar cómo actuar frente a él con un nivel alto de seguridad.

9. Bibliografía

[1] Unión Europea. Directiva (UE) 2004/54/CE del Parlamento Europeo y del Consejo, de 29 de abril de 2004, sobre requisitos mínimos de seguridad para túneles de la red transeuropea de carreteras. Disponible en: https://eur-lex.europa.eu/legal-content/ES/TXT/PDF/?uri=CELEX:32004L0054&from=ES

[2] Sick.com. (s.f.). *Incremental encoders | DFS60 | SICK*. [Online] Disponible en: https://www.sick.com/de/en/encoders/incremental-encoders/dfs60/c/g244428 [Acceso 10 Sep. 2019].

[3] Sick.com. (s.f.). *LMS200-30106 | Soluciones de medición y detección | SICK*. [Online] Disponible en: https://www.sick.com/es/es/soluciones-de-medicion-y-deteccion/sensores-2d-lidar/lms2xx/lms200-30106/p/p109843 [Acceso 10 Sep. 2019].

[4] *Gracia Sola, Carlos. (Mayo 2013). Modelado y simulación de un robot Robucar TT*. [Trabajo Fin de Carrera] Universidad de Zaragoza. (Página 6). Disponible en: http://zaguan.unizar.es/record/10425/files/TAZ-PFC-2013-193.pdf

[5] Sick.com. (s.f.). *LMS291-S05 | Detection and ranging solutions SICK*. [Online] Disponible en: https://www.sick.com/ag/en/detection-and-ranging-solutions/2d-lidar-sensors/lms2xx/lms291-s05/p/p109849 [Acceso 10 Sep. 2019].

[6] *MobileRobots Inc. (Enero 2006). Pioneer 3 Operations Manual* [Archivo PDF] (Página 13) Disponible en: https://www.inf.ufrgs.br/~prestes/Courses/Robotics/manual_pioneer.pdf

[7] Es.mathworks.com. (s.f.). *MathWorks - Creadores de MATLAB y Simulink - MATLAB y Simulink*. [Online] Disponible en: https://es.mathworks.com [Acceso 15 Jul. 2019].

[8] Docs.scipy.org. (s.f.). *Numpy and Scipy Documentation — Numpy and Scipy documentation*. [Online] Disponible en: https://docs.scipy.org/doc/ [Acceso 29 Jul. 2019].

[9] Ros.org. (s.f.). *ROS.org | Powering the world's robots*. [Online] Disponible en: https://www.ros.org/ [Acceso 4 Jun. 2019].

[10] Wiki.ros.org. (s.f.). *Documentation - ROS Wiki*. [Online] Disponible: http://wiki.ros.org/ [Acceso 3 Nov. 2019].

[11] Gazebosim.org. (s.f.). *Gazebo : Tutorials*. [Online] Disponible en: http://gazebosim.org/tutorials [Acceso 29 Ago. 2019].

[12] Moore, T. (2016). *robot_localization wiki — robot_localization 2.6.5 documentation*. [Online] Docs.ros.org. Disponible en: http://docs.ros.org/melodic/api/robot_localization/html/index.html [Acceso 5 Nov. 2019].

[13] Wiki.ros.org. (s.f.). *robot_pose_ekf - ROS Wiki*. [Online] Disponible en: http://wiki.ros.org/robot_pose_ekf [Accesso 10 Nov. 2019].

ANEXOS

Anexo A. Algoritmos en MATLAB

El script principal utilizado es el siguiente:

```
close all
%clear all
warning('OFF')
format shortg
g2r=pi/180;
x = sym('x');

%Parámetros de ajuste
discond=5;  % Distancia para considerar discontinuidad y segmentar
            % sensible a este parámetro, si pequeño->muchas
particiones
            % con valores más bajos no reconoce alguna galería rara
desorient=4; % Desorientación del robot en grados para realinear los
puntos con los ejes
grados=9; % Puntos, en grados, que se eliminan
width=5.5; % Anchura del tunel
npun=4;  % Número mínimo de puntos soporte para considerar galería
(segundo segmento)
mindis=6; % Tamaño mínimo de hueco para validar una galería (el tamaño
mínimo para considerar galería es la anchura de la más pequeña que es
6.3m)
veces_repe=10; % Mínimo número de repeticiones de galeria potencial
para filtrar (validar galería) en función de la velocidad
back=0; % Elige si empiezas a la ida o a la vuelta
dissiggal=45; % Distancia para saltar la validación de la galería y no
se repita la misma
resolucion=170; % Cuantos puntos por láser hay, 360 o 180 normalmente
xR1L=[3.08,0.0,0.0]; % Localización del láser respecto del centro de
cálculo en el robot
ensanchamiento=0.5; % Nivel de ensanchamiento que puede haber en el
túnel o en el que se puede mover el robot para calcular la Y en
funcion de la referencia inicial
paralelismo=0.01; % Nivel de paralelismo que quieres que haya entre
las rectas
rangomax=30.0; % Rango máximo del laser
totalgal=12; % Número total de galerías a reconocer
minpend=0.3; % Mínima pendiente para considerar galeria
filpend=0.7; % Margen para filtrar rectas clave respecto de la mínima
pendiente
salto=0.1; % Diferencia que puede haber entre la Y y tita actual y
anterior

% Lectura de puntos láser en cartesianas y polares del túnel
% puntos=[x y d tita]
carga_datos2;
odom_nueva=odom;  % odometría actualizada con reset en galerias

figure(1)

vecesgal=0; % veces de persistencia para confirmar galería
num_laseres=size(ranges,1);
```

```
primero=6600; % Primer láser a analizar 2, ya que los bucles se
compara con el anterior
fin=num_laseres; % Último láser

galerias=[];

if back==0
    siguiente=0;
    numgal=1;   % primera galería
else
    siguiente=odom(primero,1)+5000; %Se le suman 5000m porque la
odometría puede ser muy mala
    numgal=totalgal;
end

puntosdib=[];

hayrecta=0;
galeriavalidada=0;
nlaservalidado=0;

WxL=[0.0,0.0,0.0];
ylocales=[];
headings=[];
con=0;
ok=0;
contright=0;
galdcha=0;
galeria1=[];

rectaaux=[];
segmentaux=[];
rclave=[];
rectaclave=[];
segmentclave=[];
rectaparalela=[];

pcorte=[];
ylocal=0.0;
heading=0.0;
pdcha=0.0;
pizda=0.0;
unavez=1;
difdcha=0.0;
difizda=0.0;
headingANTd=0.0;
headingANTi=0.0;
ylocANT=0.0;
headingANT=0.0;
puntosdibx=[];
puntosdiby=[];
g=[];
galmapa=[0.0 0.0];

for nlaser=primero:fin
% Obtiene los puntos del laser filtrados a 25m
  puntos=tunelLaser(nlaser,ranges,angles);

% Elimina puntos de suelo (entre +-9 grados) mediante dos condiciones.
    puntosfilw=elimina_suelo2(puntos,grados,width);
```

```
    puntosfil=elimina_suelo3(puntosfilw,grados,width);

% Gráfica de puntos filtrados como * rojos
    plot(puntosfil(:,1),puntosfil(:,2),'r*');
    axis([0 25 -25 25])

% Segmenta puntos cuando distancia en el hueco es >discond
    [nseg,fins]=segmenta(puntosfil,discond,resolucion);

% Calcula los puntos soporte de los segmentos,
% las ecuaciones de las rectas, los extremos de los segmentos,
% y la orientación del robot (heading) respecto a la pared derecha
% ylocal: coordenada "y" medida a partir de la distancia a la pared
    [segment,recta] = calcularectas6(fins,puntosfil,1);

[heading,ylocal,WxL,pdcha,pizda,con,difdcha,difizda,headingANTd,headin
gANTi,ylocANT,headingANT] =
calculaYtita(segment,recta,puntos,ylocal,heading,WxL,pdcha,pizda,unave
z,ensanchamiento,con,difdcha,difizda,paralelismo,xR1L(1),headingANTd,h
eadingANTi,ylocANT,headingANT,salto,rangomax);
    if pdcha ~= 0.0 && pizda ~= 0.0
        unavez=0;
    end

% Usamos el heading relativo para la realineacion de puntos, si esta
de vuelta el heading absoluto lo usamos para la localización
    if back==1 && abs(heading)<(pi-0.2);
        headingabs=norm_pi(heading+pi);
    end
    if back==0
        headingabs=heading;
    end

% Reorienta los puntos para alinearlos con los ejes de referencia
% de acuerdo al "heading" del robot (desorientación) y cuando la
desalineacion
% es mayor que "desorient"
  puntosfil=realinea_puntos(puntosfil,heading,desorient);

% Calcula si hay galería y donde
% Hay galeria si el segmento siguiente tiene pendiente superior al
parámetro ajustable y si el numero de puntos soporte suyo es
significativo (>npun)

[galeria,left,rectafin,segmentfin]=calculagaleria4(puntosfil,recta,seg
ment,fins,npun,mindis,nlaser,numgal,totalgal,minpend,filpend);

% Calcula el lado en el que esta la galeria

[galdcha,contright]=ladogaleria(left,contright,galeria,galdcha,veces_r
epe);

% Busca la recta clave de la galeria

[rectaclave,rclave,segmentclave,nlaservalidado,hayrecta]=calcula_recta
clave(rectafin,segmentfin,hayrecta,segmentclave,rclave,rectaclave,nlas
ervalidado);

% Cálculo de la nueva odometría a partir de la relativa de 1 paso.
    TAO1=hom(odom_nueva(nlaser-1,:));
```

```
    xWR=actxWR(TAO1,odom_rel,nlaser,WxL,headingabs);
    odom_nueva(nlaser,:)=xWR;

% Detecta si hay galeria
    [haygaleria,vecesgal]=hay_galeria(galeria,vecesgal);

% Validación de galeria

[galeriavalidada,galmapa,siguiente,vecesgal,puntosdibx,puntosdiby,gale
ria1]=validacion_galeria(puntosfil,galmapa,galeriavalidada,haygaleria,
vecesgal,back,dissiggal,veces_repe,siguiente,odom_nueva,numgal,nlaser,
puntosdibx,puntosdiby,galeria,galeria1);

% Calcula recta paralela

[rectaparalela,galdcha]=calcula_rectaparalela(galeriavalidada,galeria,
rectaclave,galdcha,recta,segment,paralelismo);

% Relocalización con la galería

[g,numgal,ok,odom_nueva,puntosdib,galerias,pcorte]=x_clave_relocalizac
ion(g,galerias,galeria1,puntosdib,pcorte,galeriavalidada,rectaclave,rc
lave,rectaparalela,galmapa,back,odom_nueva,odom_rel,odom,numgal,nlaser
validado,nlaser,puntosdibx,puntosdiby,ok);
    if ok==1
            galeriavalidada=0;
            rectaaux=[];
            rclave=[];
            rectaclave=[];
            rectaparalela=[];
            hayrecta=0;
            ok=0;
    end

% Calculo de galería si no encuentra una recta paralela antes de
terminar con la simulación

[g,numgal,odom_nueva,puntosdib,galerias,pcorte,galeriavalidada,galdcha
,rectaaux,segmentaux,rclave,rectaclave,rectaparalela,segmentclave,hayr
ecta]=x_clave_relocalizacion_sin_paralela(g,galerias,galeria1,galdcha,
puntosdib,pcorte,galeriavalidada,galmapa,back,odom_nueva,odom_rel,odom
,numgal,nlaservalidado,nlaser,fin,segmentclave,puntosdibx,puntosdiby,r
ectaaux,segmentaux,rclave,rectaclave,rectaparalela,hayrecta);

if numgal==(totalgal+1) && back==0 && nlaser==fin
    back=1;
    numgal=totalgal;
end

if numgal==0 && back==1 && nlaser==fin
    back=0;
    numgal=1;
end

end

if ~isempty(puntosdib)
    figure(2)
    plot(g(:,2),g(:,3),'bs')
    hold on
```

```
    plot(puntosdib(:,1),puntosdib(:,2),'r*')
    figure(3)
    subplot(211)
    plot(odom_nueva(:,1),odom_nueva(:,2))
    xlabel('x'); ylabel('y')
    title('odometria corregida')
    subplot(212)
    plot(odom_nueva(:,1),cos(odom_nueva(:,3)))
    xlabel('x'); ylabel('tita')
    figure(4)
    subplot(211)
    plot(odom(:,1),odom(:,2))
    xlabel('x'); ylabel('y')
    title('odometria original')
    subplot(212)
    plot(odom(:,1),cos(odom(:,3)))
    xlabel('x'); ylabel('tita')
    figure(5)
    plot(pcorte(:,1),pcorte(:,2),'bo')
    xlabel('x'); ylabel('y')
    title('puntos de corte')

end
% guarda la odometria corregida y la original
save('odometria_corregida.mat','odom_nueva')
% guarda datode galerias, puntos brutos y puntos filtrados en cada
galería
save('puntos_galeria.mat','g','puntos','puntosfil')
```

El resto de algoritmos se encuentran en el siguiente link: https://drive.google.com/open?id=1651zDLDf0YfE7FiESHnKzGuwssIS9GT2

Anexo B. Algoritmos en ROS

El script principal se muestra en lo siguiente:

```
#! /usr/bin/env python

import rospy
from sensor_msgs.msg import LaserScan
from geometry_msgs.msg import PoseStamped
from nav_msgs.msg import Odometry
from std_msgs.msg import String
import toolsLaser as tl
import toolsStraightLines as tsl
import toolsLocation as tloc
import toolsGallery as tg
import math
import numpy as np
import matplotlib.pyplot as plt
import message_filters
from tf.transformations import quaternion_from_euler

pub = rospy.Publisher('gal', String, queue_size=1)
pub2 = rospy.Publisher('new_odom', Odometry, queue_size=1)

#Parametros
grados=9.0 # Puntos, en grados, que se eliminan
width=5.0 # Anchura del tunel
discond=5.0 # Distancia para considerar discontinuidad y segmentar
sensible a este parámetro, si pequeño->muchas particiones
resolucion=340 # Cuantos puntos por láser hay, 360 o 180
normalmente
rangomax=30.0 # Rango máximo del laser
back=0 # Elegir ida o vuelta
desorient=4.0 # Desorientación del robot en grados para realinear
los puntos con los ejes
npun=15 # Número mínimo de puntos soporte para considerar galería
(segundo segmento)
mindis=6.0 # Tamaño mínimo de hueco para validar una galería (el
tamaño mínimo para considerar galería es la anchura de la más
pequeña que es 6.3m)
totalgal=6 #12 # Numero de galerias a reconocer
veces_repe=10 # Numero de repeticiones para considerar galeria
dissiggal=35.0 # Distancia para saltar la validación de la galería
y no se repita la misma
xR1L=[0.15,0.0,0.0] # Localización del láser respecto del centro de
cálculo en el robot
ensanchamiento=0.5 # Nivel de ensanchamiento que puede haber en el
túnel o en el que se puede mover el robot para calcular la Y en
funcion de la referencia inicial
paralelismo=0.01 # Nivel de paralelismo que quieres que haya entre
las rectas
minpend=0.2 # Mínima pendiente para considerar galeria
salto=0.05 # Diferencia que puede haber entre la Y y tita actual y
anterior
filpend=0.2 # Margen para filtrar rectas clave respecto de la
mínima pendiente

firstTime=1
```

```
firstTime2=1
ylocal=0.0
heading=0.0
WxL=[0.0,0.0,0.0]
headingabs=0.0
galdcha=0
contright=0
pdcha=0.0
pizda=0.0
unavez=1

clave=0
hayrecta=0
ylocales=[]
headings=[]
segmentclave=[]
rclave=[]
rectaclave=[]
rectaaux=[]
nlaservalidado=0
contright=0

vecesgal=0
galeriavalidada=0
galmapa=[]
x_punto_clave=-1.0
g=[]
pcorte=[]

odom=[]
odom_rel=[]
odom_nueva=[]
i=-1
ok=0
xWR=[]
con=0
difdcha=0.0
difizda=0.0
headingANTd=0.0
headingANTi=0.0
ylocANT=0.0
headingANT=0.0

x=[]
y=[]
tita=[]
xc=[]
yc=[]
titac=[]

printea=1

n_odom = Odometry()

def callback(msg, msg2):
        global firstTime, firstTime2, ylocal, heading, headingabs,
numgal, galdcha, contright, clave, hayrecta, ylocales, headings,
segmentclave, rclave, rectaclave, nlaservalidado, vecesgal,
```

```
siguiente, numgal, galeriavalidada, odom, odom_rel, odom_nueva,
galmapa, i, x_punto_clave, xWR, rectaaux, ok, WxL, pdcha, pizda,
unavez, con, difdcha, difizda, headingANTi, headingANTd, ylocANT,
x, y, xc, yc, contright, printea, odom_filtro_kalman, xk, yk, tita,
titac, titak, headingANT, odom_yth, g, pcorte
        global grados, width, discond, resolucion, rangomax, back,
desorient, npun, mindis, veces_repe, dissiggal, xR1L,
ensanchamiento, paralelismo, minpend, n_odom, salto, filpend

        #Extraccion de la informacion del topic
        loc=[msg2.pose.position.x,msg2.pose.position.y,msg2.pose.pos
ition.z,msg2.pose.orientation.x,msg2.pose.orientation.y,msg2.pose.o
rientation.z,msg2.pose.orientation.w]
        odo=tloc.odometriaabs(loc)
        odom.append(odo)
        x.append(odom[-1][0])
        y.append(odom[-1][1])
        tita.append(math.cos(odom[-1][2]))

        odorel=tloc.odometriarel2(odom)
        odom_rel.append(odorel)
        i=i+1

        if len(odom) == 2:
                odomdepaso=odom[0]
                odomdepaso2=odom[1]
                odom_nueva.append(odomdepaso)
                odom_nueva.append(odomdepaso2)
                odom_yth.append(odomdepaso)
                odom_yth.append(odomdepaso2)
                xc.append(odomdepaso[0])
                xc.append(odomdepaso2[0])
                yc.append(odomdepaso[1])
                yc.append(odomdepaso2[1])
                titac.append(math.cos(odomdepaso[2]))
                titac.append(math.cos(odomdepaso2[2]))

        #Inicializacion de parametros en funcion de ida o vuelta
        if firstTime==1:
                if back == 0:
                        siguiente=0
                        numgal=0
                else:
                        siguiente=odom[-1][0]+5000
                        numgal=totalgal
                firstTime=0


        angles=tl.allangles(msg.angle_min,msg.angle_max,msg.angle_in
crement)

        #Filtracion puntos lejanos
        points=tl.pointsFilter(msg.ranges,angles)

        #Filtracion puntos pertenecientes al suelo
         pointsfilw=tl.eliminasuelo1(points,grados,width)
         pointsfil=tl.eliminasuelo2(pointsfilw,grados,width)
```

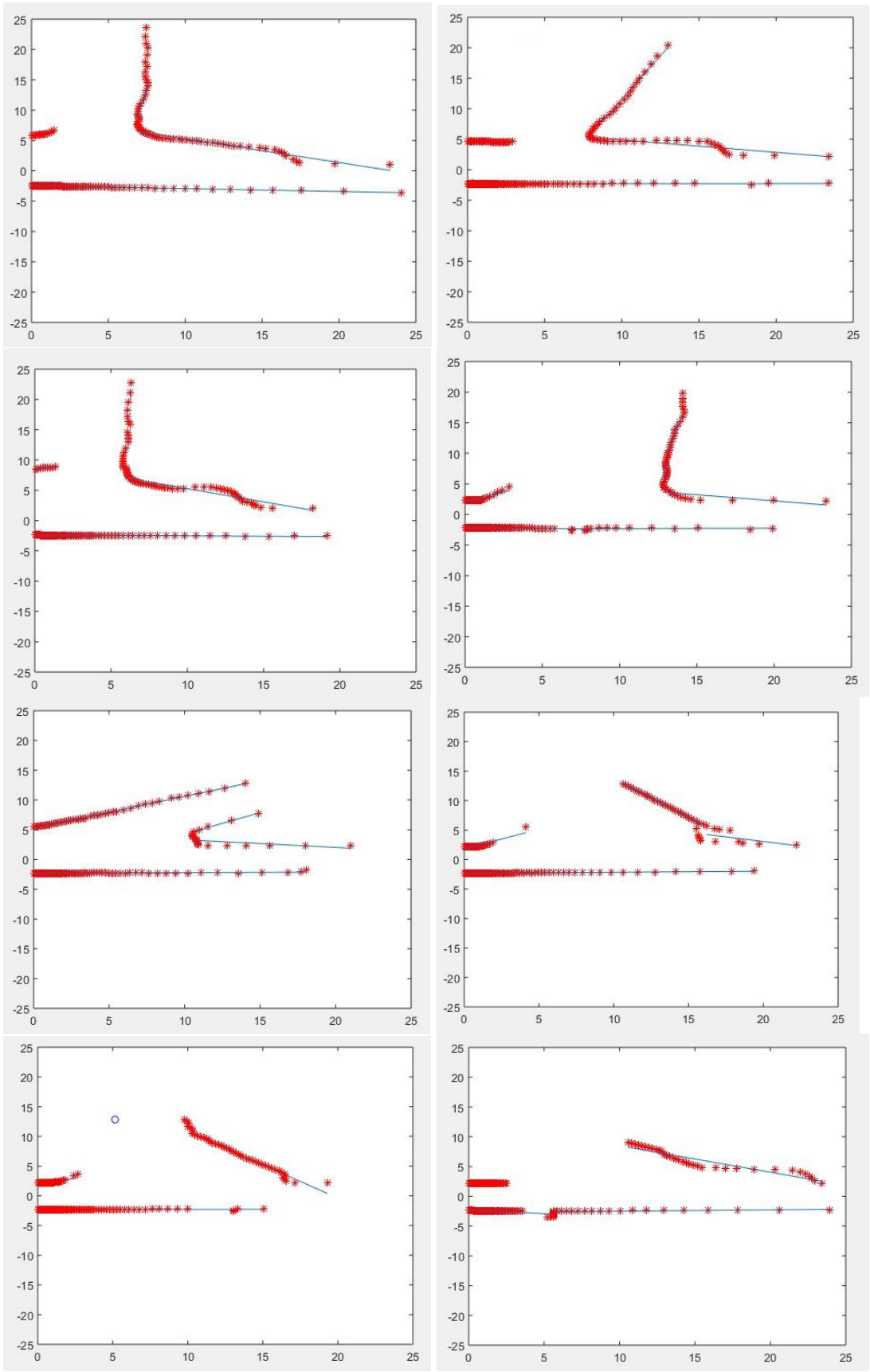
```
        #Segmentacion de puntos
         fins=tl.segmenta(pointsfil,discond,resolucion)

        #Calculo de rectas
        segment,recta=tsl.calcularectas6(fins,pointsfil)
        heading,ylocal,WxL,pdcha,pizda,con,difdcha,difizda,headingAN
Ti,headingANTd,ylocANT,headingANT=tsl.calculaYtita(segment,recta,po
intsfil,ylocal,heading,WxL,pdcha,pizda,unavez,ensanchamiento,rangom
ax,con,difdcha,difizda,paralelismo,xR1L[0],headingANTi,headingANTd,
ylocANT,headingANT,salto)

         if pdcha!=0.0 and pizda !=0.0:
               unavez=0

        #Calculo del heading absoluto
         headingabs=tloc.calculaheadingabs(heading,back)

        #Realineacion de puntos
         pointsfil=tl.realinea_puntos(pointsfil,heading,desorient)

        #Calculo de galerias
        galeria,left,rectafin,segmentfin=tg.calculagaleria(pointsfil
,recta,segment,fins,npun,mindis,i,numgal,totalgal,minpend,filpend)

        #Calculo del lado de la galeria

galdcha,contright=tg.ladogaleria(left,galeria,contright,galdcha)

        #Busqueda de la recta clave
        rectaclave,rclave,segmentclave,nlaservalidado,ylocales,headi
ngs,hayrecta,clave=tg.buscarectaclave(rectafin,segmentfin,ylocal,he
adingabs,hayrecta,clave,ylocales,headings,segmentclave,rclave,recta
clave,nlaservalidado)

        #Actualizacion de la odometria relativa en 1 paso
        if odom_nueva != []:
               if firstTime2==1:

                      TAO1=tloc.hom([odom_nueva[-2][0],odom_nueva[-
2][1],odom_nueva[-2][2]])

                      xWR=tloc.actxWR(TAO1,odom_rel,WxL,headingabs)

                      odom_nueva.pop() #Elimino el ultimo elemento
                      odom_nueva.append(xWR)
                      odom_yth.pop() #Elimino el ultimo elemento
                      odom_yth.append(xWR)

                      xc.pop()
                      yc.pop()
                      titac.pop()
                      xc.append(odom_nueva[-1][0])
                      yc.append(odom_nueva[-1][1])
                      titac.append(math.cos(odom_nueva[-1][2]))
                      firstTime2=0

               else:
```

```
                        TAO1=tloc.hom([odom_nueva[-1][0],odom_nueva[-
1][1],odom_nueva[-1][2]])

                        xWR=tloc.actxWR(TAO1,odom_rel,WxL,headingabs)

                        odom_nueva.append(xWR)
                        odom_yth.append(xWR)

                         xc.append(odom_nueva[-1][0])
                        yc.append(odom_nueva[-1][1])
                        titac.append(math.cos(odom_nueva[-1][2]))

        #Deteccion de galeria
        haygaleria,vecesgal=tg.hay_galeria_o_hueco(galeria,vecesgal)
        galeriavalidada,galmapa,siguiente,vecesgal=tg.validacion_gal
eria(galmapa,galeriavalidada,haygaleria,vecesgal,back,dissiggal,vec
es_repe,siguiente,odom_nueva,numgal)

        #Busqueda de recta paralela para interseccion
        rectaparalela,repa,galdcha=tg.buscarectaparalela(paralelismo
,galeriavalidada,galeria,rectaclave,galdcha,recta,segment)

        #Calculo de la intersección y relocalizacion
        g,pcorte,odom_nueva,numgal,
ok,xc,yc,titac=tg.x_clave_relocalizacion(galeriavalidada,g,pcorte,r
ectaclave,rclave,repa,rectaparalela,x_punto_clave,xWR,galmapa,yloca
les,headings,back,odom_nueva,odom,odom_rel,numgal,nlaservalidado,ok
,xc,yc,titac)

        if ok==1: # Reset de variables
                galeriavalidada=0
                rectaaux=[]
                rclave=[]
                rectaclave=[]
                rectaparalela=[]
                clave=0
                hayrecta=0
                ylocales=[]
                headings=[]
                ok=0


        if len(g)==printea:
                print g
                print pcorte
                mensaje = "Num, xc, yc, titac, nlaser, x, y, tita:
%s" % g
                #Publicacion del topic
                pub.publish(mensaje)

                printea=printea+1

        if len(g)==totalgal:

                plt.figure(1)
                plt.subplot(211)
                plt.plot(xc,yc, 'bo')
                plt.xlabel('X corregida')
```

```
                plt.ylabel('Y corregida')

                plt.subplot(212)
                plt.plot(x,y, 'ro')
                plt.xlabel('X')
                plt.ylabel('Y')

                plt.figure(2)
                plt.subplot(211)
                plt.plot(xc,titac, 'bo')
                plt.xlabel('X corregida')
                plt.ylabel('cos(tita) corregida')

                plt.subplot(212)
                plt.plot(x,tita, 'ro')
                plt.xlabel('X')
                plt.ylabel('cos(tita)')
                plt.show()

        if len(odom_nueva)!=0:
                n_odom.header.stamp = msg2.header.stamp
                n_odom.header.frame_id = msg2.header.frame_id
                n_odom.child_frame_id = "base_link"
                n_odom.pose.pose.position.x = odom_nueva[-1][0]
                n_odom.pose.pose.position.y = odom_nueva[-1][1]
                n_odom.pose.pose.position.z = 0.0

                q_new_odom = quaternion_from_euler (0.0,
0.0,odom_nueva[-1][2])

                n_odom.pose.pose.orientation.x = q_new_odom[0]
                n_odom.pose.pose.orientation.y = q_new_odom[1]
                n_odom.pose.pose.orientation.z = q_new_odom[2]
                n_odom.pose.pose.orientation.w = q_new_odom[3]
                #Publicacion del topic
                pub2.publish(n_odom)


rospy.init_node('my_scan')
#Suscripcion a los topics del laser y localización, sincronizados
sub = message_filters.Subscriber('/laser/scan', LaserScan)
sub_2 = message_filters.Subscriber('/loc', PoseStamped)
ts = message_filters.ApproximateTimeSynchronizer([sub, sub_2], 1,
0.1, allow_headerless=True)
ts.registerCallback(callback)
rospy.spin()
```

El resto de algoritmos se encuentran en el siguiente link: https://drive.google.com/open?id=1zfJ9XdkOVl-Dyzu7tK-4DuKnwMh-4UgT

Anexo C. Galerías del túnel de Somport



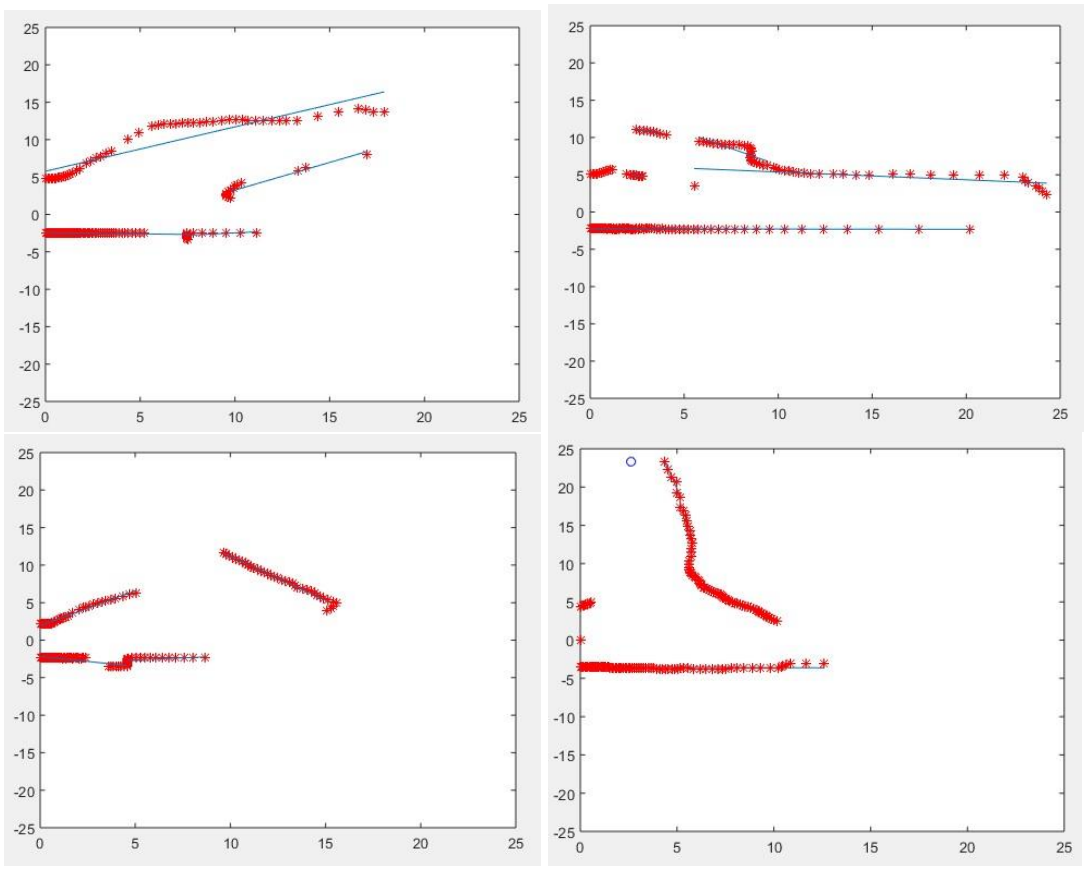


Figura 44. Galerías, de izquierda a derecha, de la 1 a la 12.

Anexo D. Escenarios en Gazebo

El modelo del túnel principal, *wall_box_unit,* se puede ver en el siguiente extracto del fichero del modelo:

Cada link es un cubo modificado donde se le indican la posición en la que va a estar situado (pose frame) y el tamaño, y por consiguiente su geometría (size). El resto de parámetros no son relevantes para estos modelos. Como ejemplo, vamos a explicar el *link_0*: Se trata de un cubo modificado de tal forma que tiene 20 metros de longitud, 10 centímetros de grosor y 1 metro de altitud. Tiene su centro de referencia en X = 10 metros e Y = 0.05 metros. Por tanto deducimos que respecto de la referencia del 'mundo' en Gazebo, va a estar a 10 metros del origen respecto de X, y a 2,5 metros del origen respecto de Y.

```
<?xml version='1.0'?>
<sdf version='1.6'>
  <model name='wall_box_unit'>
    <link name='link_0'>
      <pose frame=''>20.0 2.55 0.5 0 0 0</pose>
      <inertial>
        <mass>1</mass>
        <inertia>
          <ixx>0.166667</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.166667</iyy>
          <iyz>0</iyz>
          <izz>0.166667</izz>
        </inertia>
      </inertial>
      <collision name='collision'>
        <geometry>
          <box>
            <size>20 0.10 1</size>
          </box>
        </geometry>
        <max_contacts>10</max_contacts>
      </collision>
      <visual name='visual'>
        <geometry>
          <box>
            <size>20 0.10 1</size>
          </box>
        </geometry>
        <material>
          <script>
            <name>Gazebo/Grey</name>
            <uri>file://media/materials/scripts/gazebo.material</uri>
          </script>
```

```
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
</link>

<link name='link_1'>
  <pose frame=''>74.0 -2.55 0.5 0 0 0</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>128 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>128 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
</link>

<link name='link_2'>
```

```
  <pose frame=''>37.071067812 9.621067812 0.5 0 0 0.79</pose>
  <inertial>
   <mass>1</mass>
   <inertia>
    <ixx>0.166667</ixx>
    <ixy>0</ixy>
    <ixz>0</ixz>
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
    <izz>0.166667</izz>
   </inertia>
  </inertial>
  <collision name='collision'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <material>
    <script>
     <name>Gazebo/Grey</name>
     <uri>file://media/materials/scripts/gazebo.material</uri>
    </script>
   </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_3'>
  <pose frame=''>45.071067812 9.621067812 0.5 0 0 0.79</pose>
  <inertial>
   <mass>1</mass>
   <inertia>
    <ixx>0.166667</ixx>
    <ixy>0</ixy>
    <ixz>0</ixz>
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
```

```
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_4'>
  <pose frame=''>88.0 2.55 0.5 0 0 0</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>100 0.10 1</size>
      </box>
    </geometry>
```

```
      <max_contacts>10</max_contacts>
     </collision>
     <visual name='visual'>
      <geometry>
       <box>
        <size>100 0.10 1</size>
       </box>
      </geometry>
      <material>
       <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
       </script>
      </material>
     </visual>
     <self_collide>0</self_collide>
     <enable_wind>1</enable_wind>
     <kinematic>0</kinematic>
     <gravity>0</gravity>
    </link>

<!-- primer tramo de tunel gal 30-38 izda -->

    <link name='link_5'>
     <pose frame=''>148.0 -2.55 0.5 0 0 0</pose>
     <inertial>
      <mass>1</mass>
      <inertia>
       <ixx>0.166667</ixx>
       <ixy>0</ixy>
       <ixz>0</ixz>
       <iyy>0.166667</iyy>
       <iyz>0</iyz>
       <izz>0.166667</izz>
      </inertia>
     </inertial>
     <collision name='collision'>
      <geometry>
       <box>
        <size>20 0.10 1</size>
       </box>
      </geometry>
      <max_contacts>10</max_contacts>
     </collision>
     <visual name='visual'>
      <geometry>
       <box>
        <size>20 0.10 1</size>
       </box>
```

```
   </geometry>
   <material>
    <script>
     <name>Gazebo/Grey</name>
     <uri>file://media/materials/scripts/gazebo.material</uri>
    </script>
   </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

 <link name='link_6'>
  <pose frame=''>201.5 2.55 0.5 0 0 0</pose>
  <inertial>
   <mass>1</mass>
   <inertia>
    <ixx>0.166667</ixx>
    <ixy>0</ixy>
    <ixz>0</ixz>
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
    <izz>0.166667</izz>
   </inertia>
  </inertial>
  <collision name='collision'>
   <geometry>
    <box>
     <size>127 0.10 1</size>
    </box>
   </geometry>
   <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
   <geometry>
    <box>
     <size>127 0.10 1</size>
    </box>
   </geometry>
   <material>
    <script>
     <name>Gazebo/Grey</name>
     <uri>file://media/materials/scripts/gazebo.material</uri>
    </script>
   </material>
  </visual>
  <self_collide>0</self_collide>
```

```
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

 <link name='link_7'>
  <pose frame=''>150.9289322 -9.621067812 0.5 0 0 0.79</pose>
  <inertial>
   <mass>1</mass>
   <inertia>
    <ixx>0.166667</ixx>
    <ixy>0</ixy>
    <ixz>0</ixz>
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
    <izz>0.166667</izz>
   </inertia>
  </inertial>
  <collision name='collision'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <material>
    <script>
     <name>Gazebo/Grey</name>
     <uri>file://media/materials/scripts/gazebo.material</uri>
    </script>
   </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_8'>
  <pose frame=''>157.9289322 -9.621067812 0.5 0 0 0.79</pose>
  <inertial>
   <mass>1</mass>
```

```
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_9'>
  <pose frame=''>215.0 -2.55 0.5 0 0 0</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
```

```
      <collision name='collision'>
       <geometry>
        <box>
         <size>100 0.10 1</size>
        </box>
       </geometry>
       <max_contacts>10</max_contacts>
      </collision>
      <visual name='visual'>
       <geometry>
        <box>
         <size>100 0.10 1</size>
        </box>
       </geometry>
       <material>
        <script>
         <name>Gazebo/Grey</name>
         <uri>file://media/materials/scripts/gazebo.material</uri>
        </script>
       </material>
      </visual>
      <self_collide>0</self_collide>
      <enable_wind>1</enable_wind>
      <kinematic>0</kinematic>
      <gravity>0</gravity>
     </link>

<!-- segundo tramo de tunel gal 158-165 dcha -->

     <link name='link_10'>
      <pose frame=''>265 2.55 0.5 0 0 0</pose>
      <inertial>
       <mass>1</mass>
       <inertia>
        <ixx>0.166667</ixx>
        <ixy>0</ixy>
        <ixz>0</ixz>
        <iyy>0.166667</iyy>
        <iyz>0</iyz>
        <izz>0.166667</izz>
       </inertia>
      </inertial>
      <collision name='collision'>
       <geometry>
        <box>
         <size>20 0.10 1</size>
        </box>
       </geometry>
       <max_contacts>10</max_contacts>
```

```
 </collision>
 <visual name='visual'>
  <geometry>
   <box>
    <size>20 0.10 1</size>
   </box>
  </geometry>
  <material>
   <script>
    <name>Gazebo/Grey</name>
    <uri>file://media/materials/scripts/gazebo.material</uri>
   </script>
  </material>
 </visual>
 <self_collide>0</self_collide>
 <enable_wind>1</enable_wind>
 <kinematic>0</kinematic>
 <gravity>0</gravity>
</link>

<link name='link_11'>
 <pose frame=''>318.5 -2.55 0.5 0 0 0</pose>
 <inertial>
  <mass>1</mass>
  <inertia>
   <ixx>0.166667</ixx>
   <ixy>0</ixy>
   <ixz>0</ixz>
   <iyy>0.166667</iyy>
   <iyz>0</iyz>
   <izz>0.166667</izz>
  </inertia>
 </inertial>
 <collision name='collision'>
  <geometry>
   <box>
    <size>127 0.10 1</size>
   </box>
  </geometry>
  <max_contacts>10</max_contacts>
 </collision>
 <visual name='visual'>
  <geometry>
   <box>
    <size>127 0.10 1</size>
   </box>
  </geometry>
  <material>
   <script>
```

```
      <name>Gazebo/Grey</name>
      <uri>file://media/materials/scripts/gazebo.material</uri>
     </script>
    </material>
   </visual>
   <self_collide>0</self_collide>
   <enable_wind>1</enable_wind>
   <kinematic>0</kinematic>
   <gravity>0</gravity>
  </link>

  <link name='link_12'>
   <pose frame=''>267.9289322 9.621067812 0.5 0 0 -0.79</pose>
   <inertial>
    <mass>1</mass>
    <inertia>
     <ixx>0.166667</ixx>
     <ixy>0</ixy>
     <ixz>0</ixz>
     <iyy>0.166667</iyy>
     <iyz>0</iyz>
     <izz>0.166667</izz>
    </inertia>
   </inertial>
   <collision name='collision'>
    <geometry>
     <box>
      <size>20 0.10 1</size>
     </box>
    </geometry>
    <max_contacts>10</max_contacts>
   </collision>
   <visual name='visual'>
    <geometry>
     <box>
      <size>20 0.10 1</size>
     </box>
    </geometry>
    <material>
     <script>
      <name>Gazebo/Grey</name>
      <uri>file://media/materials/scripts/gazebo.material</uri>
     </script>
    </material>
   </visual>
   <self_collide>0</self_collide>
   <enable_wind>1</enable_wind>
   <kinematic>0</kinematic>
   <gravity>0</gravity>
```

```
 </link>

<link name='link_13'>
  <pose frame=''>274.9289322 9.621067812 0.5 0 0 -0.79</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_14'>
  <pose frame=''>332 2.55 0.5 0 0 0</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
```

```
          <ixz>0</ixz>
          <iyy>0.166667</iyy>
          <iyz>0</iyz>
          <izz>0.166667</izz>
         </inertia>
        </inertial>
        <collision name='collision'>
         <geometry>
          <box>
           <size>100 0.10 1</size>
          </box>
         </geometry>
         <max_contacts>10</max_contacts>
        </collision>
        <visual name='visual'>
         <geometry>
          <box>
           <size>100 0.10 1</size>
          </box>
         </geometry>
         <material>
          <script>
           <name>Gazebo/Grey</name>
           <uri>file://media/materials/scripts/gazebo.material</uri>
          </script>
         </material>
        </visual>
        <self_collide>0</self_collide>
        <enable_wind>1</enable_wind>
        <kinematic>0</kinematic>
        <gravity>0</gravity>
       </link>

<!-- tercer tramo de tunel gal 275-282 izda -->

       <link name='link_15'>
        <pose frame=''>392 -2.55 0.5 0 0 0</pose>
        <inertial>
         <mass>1</mass>
         <inertia>
          <ixx>0.166667</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.166667</iyy>
          <iyz>0</iyz>
          <izz>0.166667</izz>
         </inertia>
        </inertial>
        <collision name='collision'>
```

```
  <geometry>
   <box>
    <size>20 0.10 1</size>
   </box>
  </geometry>
  <max_contacts>10</max_contacts>
 </collision>
 <visual name='visual'>
  <geometry>
   <box>
    <size>20 0.10 1</size>
   </box>
  </geometry>
  <material>
   <script>
    <name>Gazebo/Grey</name>
    <uri>file://media/materials/scripts/gazebo.material</uri>
   </script>
  </material>
 </visual>
 <self_collide>0</self_collide>
 <enable_wind>1</enable_wind>
 <kinematic>0</kinematic>
 <gravity>0</gravity>
</link>

<link name='link_16'>
 <pose frame=''>446 2.55 0.5 0 0 0</pose>
 <inertial>
  <mass>1</mass>
  <inertia>
   <ixx>0.166667</ixx>
   <ixy>0</ixy>
   <ixz>0</ixz>
   <iyy>0.166667</iyy>
   <iyz>0</iyz>
   <izz>0.166667</izz>
  </inertia>
 </inertial>
 <collision name='collision'>
  <geometry>
   <box>
    <size>128 0.10 1</size>
   </box>
  </geometry>
  <max_contacts>10</max_contacts>
 </collision>
 <visual name='visual'>
  <geometry>
```

```
      <box>
        <size>128 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
</link>

<link name='link_17'>
  <pose frame=''>409.0710678 -9.621067812 0.5 0 0 -0.79</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
```

```
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_18'>
  <pose frame=''>417.0710678 -9.621067812 0.5 0 0 -0.79</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_19'>
```

```
      <pose frame=''>460 -2.55 0.5 0 0 0</pose>
      <inertial>
       <mass>1</mass>
       <inertia>
        <ixx>0.166667</ixx>
        <ixy>0</ixy>
        <ixz>0</ixz>
        <iyy>0.166667</iyy>
        <iyz>0</iyz>
        <izz>0.166667</izz>
       </inertia>
      </inertial>
      <collision name='collision'>
       <geometry>
        <box>
         <size>100 0.10 1</size>
        </box>
       </geometry>
       <max_contacts>10</max_contacts>
      </collision>
      <visual name='visual'>
       <geometry>
        <box>
         <size>100 0.10 1</size>
        </box>
       </geometry>
       <material>
        <script>
         <name>Gazebo/Grey</name>
         <uri>file://media/materials/scripts/gazebo.material</uri>
        </script>
       </material>
      </visual>
      <self_collide>0</self_collide>
      <enable_wind>1</enable_wind>
      <kinematic>0</kinematic>
      <gravity>0</gravity>
     </link>

<!-- cuarto tramo de tunel gal 402-410 dcha -->

     <link name='link_20'>
      <pose frame=''>520 2.55 0.5 0 0 0</pose>
      <inertial>
       <mass>1</mass>
       <inertia>
        <ixx>0.166667</ixx>
        <ixy>0</ixy>
        <ixz>0</ixz>
```

```
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
    <izz>0.166667</izz>
   </inertia>
  </inertial>
  <collision name='collision'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <material>
    <script>
     <name>Gazebo/Grey</name>
     <uri>file://media/materials/scripts/gazebo.material</uri>
    </script>
   </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

 <link name='link_21'>
  <pose frame=''>574.1 -2.55 0.5 0 0 0</pose>
  <inertial>
   <mass>1</mass>
   <inertia>
    <ixx>0.166667</ixx>
    <ixy>0</ixy>
    <ixz>0</ixz>
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
    <izz>0.166667</izz>
   </inertia>
  </inertial>
  <collision name='collision'>
   <geometry>
    <box>
     <size>128.2 0.10 1</size>
```

```
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>128.2 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
</link>

<link name='link_22'>
  <pose frame=''>530.05 12.55 0.5 0 0 1.57</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
```

```
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_23'>
  <pose frame=''>538.15 12.55 0.5 0 0 1.57</pose>
  <inertial>
    <mass>1</mass>
    <inertia>
      <ixx>0.166667</ixx>
      <ixy>0</ixy>
      <ixz>0</ixz>
      <iyy>0.166667</iyy>
      <iyz>0</iyz>
      <izz>0.166667</izz>
    </inertia>
  </inertial>
  <collision name='collision'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
    <geometry>
      <box>
        <size>20 0.10 1</size>
      </box>
    </geometry>
    <material>
      <script>
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
      </script>
    </material>
  </visual>
  <self_collide>0</self_collide>
```

```
      <enable_wind>1</enable_wind>
      <kinematic>0</kinematic>
      <gravity>0</gravity>
    </link>

    <link name='link_24'>
      <pose frame=''>588.2 2.55 0.5 0 0 0</pose>
      <inertial>
        <mass>1</mass>
        <inertia>
          <ixx>0.166667</ixx>
          <ixy>0</ixy>
          <ixz>0</ixz>
          <iyy>0.166667</iyy>
          <iyz>0</iyz>
          <izz>0.166667</izz>
        </inertia>
      </inertial>
      <collision name='collision'>
        <geometry>
          <box>
            <size>100 0.10 1</size>
          </box>
        </geometry>
        <max_contacts>10</max_contacts>
      </collision>
      <visual name='visual'>
        <geometry>
          <box>
            <size>100 0.10 1</size>
          </box>
        </geometry>
        <material>
          <script>
            <name>Gazebo/Grey</name>
            <uri>file://media/materials/scripts/gazebo.material</uri>
          </script>
        </material>
      </visual>
      <self_collide>0</self_collide>
      <enable_wind>1</enable_wind>
      <kinematic>0</kinematic>
      <gravity>0</gravity>
    </link>

<!-- quinto tramo de tunel gal 530.1-538 izda -->

    <link name='link_25'>
      <pose frame=''>648.2 -2.55 0.5 0 0 0</pose>
```

```
 <inertial>
  <mass>1</mass>
  <inertia>
   <ixx>0.166667</ixx>
   <ixy>0</ixy>
   <ixz>0</ixz>
   <iyy>0.166667</iyy>
   <iyz>0</iyz>
   <izz>0.166667</izz>
  </inertia>
 </inertial>
 <collision name='collision'>
  <geometry>
   <box>
    <size>20 0.10 1</size>
   </box>
  </geometry>
  <max_contacts>10</max_contacts>
 </collision>
 <visual name='visual'>
  <geometry>
   <box>
    <size>20 0.10 1</size>
   </box>
  </geometry>
  <material>
   <script>
    <name>Gazebo/Grey</name>
    <uri>file://media/materials/scripts/gazebo.material</uri>
   </script>
  </material>
 </visual>
 <self_collide>0</self_collide>
 <enable_wind>1</enable_wind>
 <kinematic>0</kinematic>
 <gravity>0</gravity>
</link>

<link name='link_26'>
 <pose frame=''>702.3 2.55 0.5 0 0 0</pose>
 <inertial>
  <mass>1</mass>
  <inertia>
   <ixx>0.166667</ixx>
   <ixy>0</ixy>
   <ixz>0</ixz>
   <iyy>0.166667</iyy>
   <iyz>0</iyz>
   <izz>0.166667</izz>
```

```
  </inertia>
 </inertial>
 <collision name='collision'>
  <geometry>
   <box>
    <size>128.2 0.10 1</size>
   </box>
  </geometry>
  <max_contacts>10</max_contacts>
 </collision>
 <visual name='visual'>
  <geometry>
   <box>
    <size>128.2 0.10 1</size>
   </box>
  </geometry>
  <material>
   <script>
    <name>Gazebo/Grey</name>
    <uri>file://media/materials/scripts/gazebo.material</uri>
   </script>
  </material>
 </visual>
 <self_collide>0</self_collide>
 <enable_wind>1</enable_wind>
 <kinematic>0</kinematic>
 <gravity>0</gravity>
</link>

<link name='link_27'>
 <pose frame=''>658.25 -12.55 0.5 0 0 1.57</pose>
 <inertial>
  <mass>1</mass>
  <inertia>
   <ixx>0.166667</ixx>
   <ixy>0</ixy>
   <ixz>0</ixz>
   <iyy>0.166667</iyy>
   <iyz>0</iyz>
   <izz>0.166667</izz>
  </inertia>
 </inertial>
 <collision name='collision'>
  <geometry>
   <box>
    <size>20 0.10 1</size>
   </box>
  </geometry>
  <max_contacts>10</max_contacts>
```

```
  </collision>
  <visual name='visual'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <material>
    <script>
     <name>Gazebo/Grey</name>
     <uri>file://media/materials/scripts/gazebo.material</uri>
    </script>
   </material>
  </visual>
  <self_collide>0</self_collide>
  <enable_wind>1</enable_wind>
  <kinematic>0</kinematic>
  <gravity>0</gravity>
 </link>

<link name='link_28'>
  <pose frame=''>666.25 -12.55 0.5 0 0 1.57</pose>
  <inertial>
   <mass>1</mass>
   <inertia>
    <ixx>0.166667</ixx>
    <ixy>0</ixy>
    <ixz>0</ixz>
    <iyy>0.166667</iyy>
    <iyz>0</iyz>
    <izz>0.166667</izz>
   </inertia>
  </inertial>
  <collision name='collision'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <max_contacts>10</max_contacts>
  </collision>
  <visual name='visual'>
   <geometry>
    <box>
     <size>20 0.10 1</size>
    </box>
   </geometry>
   <material>
    <script>
```

```
        <name>Gazebo/Grey</name>
        <uri>file://media/materials/scripts/gazebo.material</uri>
       </script>
      </material>
    </visual>
    <self_collide>0</self_collide>
    <enable_wind>1</enable_wind>
    <kinematic>0</kinematic>
    <gravity>0</gravity>
   </link>

  <link name='link_29'>
    <pose frame=''>716.3 -2.55 0.5 0 0 0</pose>
    <inertial>
      <mass>1</mass>
      <inertia>
        <ixx>0.166667</ixx>
        <ixy>0</ixy>
        <ixz>0</ixz>
        <iyy>0.166667</iyy>
        <iyz>0</iyz>
        <izz>0.166667</izz>
      </inertia>
    </inertial>
    <collision name='collision'>
      <geometry>
        <box>
          <size>100 0.10 1</size>
        </box>
      </geometry>
      <max_contacts>10</max_contacts>
    </collision>
    <visual name='visual'>
      <geometry>
        <box>
          <size>100 0.10 1</size>
        </box>
      </geometry>
      <material>
        <script>
          <name>Gazebo/Grey</name>
          <uri>file://media/materials/scripts/gazebo.material</uri>
        </script>
      </material>
    </visual>
    <self_collide>0</self_collide>
    <enable_wind>1</enable_wind>
    <kinematic>0</kinematic>
    <gravity>0</gravity>
```

```
    </link>


  </model>
</sdf>
```

Como se puede ver cada link es una pieza del túnel obtenida por la modificación de la forma de un cubo. Una vez creado el modelo que como se puede observar contiene seis galerías se procede a crear un fichero de mapa:

```
<?xml version="1.0" ?>
<sdf version="1.6">
  <world name="tunnel_carol">

    <gui fullscreen='0'>
      <camera name='user_camera'>
        <pose>-6.3 -4.2 3.6 0 0.268 0.304</pose>
      </camera>
    </gui>

    <scene>
      <ambient>0.2 0.2 0.2 1.0</ambient>
      <background>0.34 0.39 0.43 1.0</background>
      <grid>false</grid>
      <origin_visual>false</origin_visual>
    </scene>

    <model name='ground_plane'>
      <static>1</static>
      <link name='link'>
        <collision name='collision'>
          <geometry>
            <plane>
              <normal>0 0 1</normal>
              <size>1500 20</size>
            </plane>
          </geometry>
          <surface>
            <friction>
              <ode>
                <mu>100</mu>
                <mu2>50</mu2>
              </ode>
            </friction>
            <bounce/>
            <contact>
              <ode/>
            </contact>
          </surface>
```

```
      <max_contacts>10</max_contacts>
     </collision>
     <visual name='visual'>
      <cast_shadows>0</cast_shadows>
      <geometry>
       <plane>
        <normal>0 0 1</normal>
        <size>1500 20</size>
       </plane>
      </geometry>
      <material>
       <script>
        <uri>file://media/materials/scripts/gazebo.material</uri>
        <name>Gazebo/Grey</name>
       </script>
      </material>
     </visual>
     <velocity_decay>
      <linear>0</linear>
      <angular>0</angular>
     </velocity_decay>
     <self_collide>0</self_collide>
     <kinematic>0</kinematic>
     <gravity>1</gravity>
    </link>
   </model>

   <include>
     <uri>model://sun</uri>
   </include>

   <include>
    <uri>model://wall_box_unit</uri>
    <name>box_1</name>
    <pose>0 0 0 0 0 0</pose>
   </include>

  </world>
</sdf>
```

En este fichero de mapa se ve la creación del escenario con su plano de suelo, luz y el modelo del mapa creado como *wall_box_unit*.

Además de escenario principal que contiene galerías tanto a derecha como a izquierda de distintas anchuras, se crearon otros modelos más pequeños con tres galerías: primero a izquierda, después a derecha y por último mezclándolas. Se muestran los mapas en las figuras siguientes. Sus correspondientes algoritmos se encuentran en el siguiente link: https://drive.google.com/open?id=1QghM2gdryMc6FaadBhx_DjD41ltGWpbX

- Galerías a izquierda:

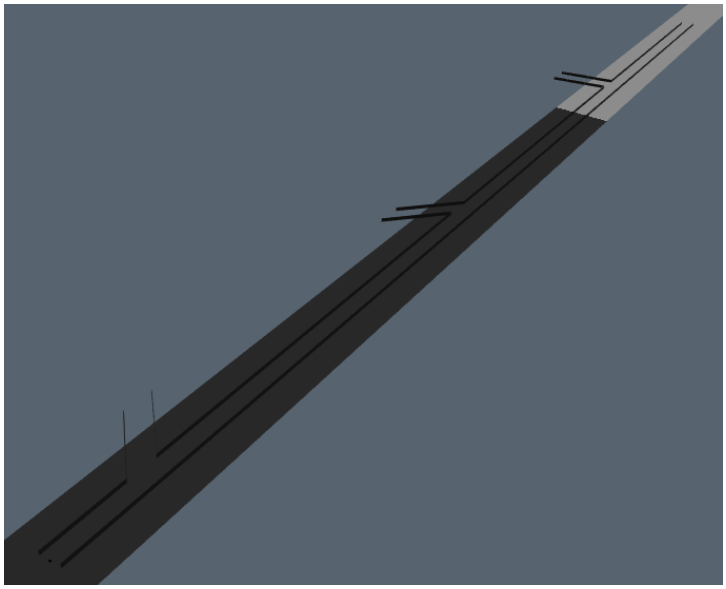


Figura 45. Túnel con galerías a la izquierda.

- Galerías a derecha:

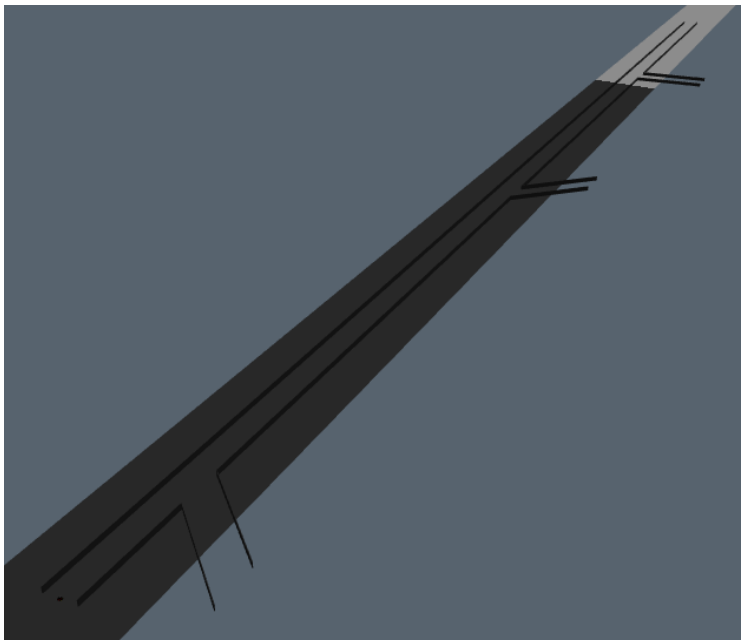


Figura 46. Túnel con galerías a la derecha.

- Galerías a izquierda y derecha:

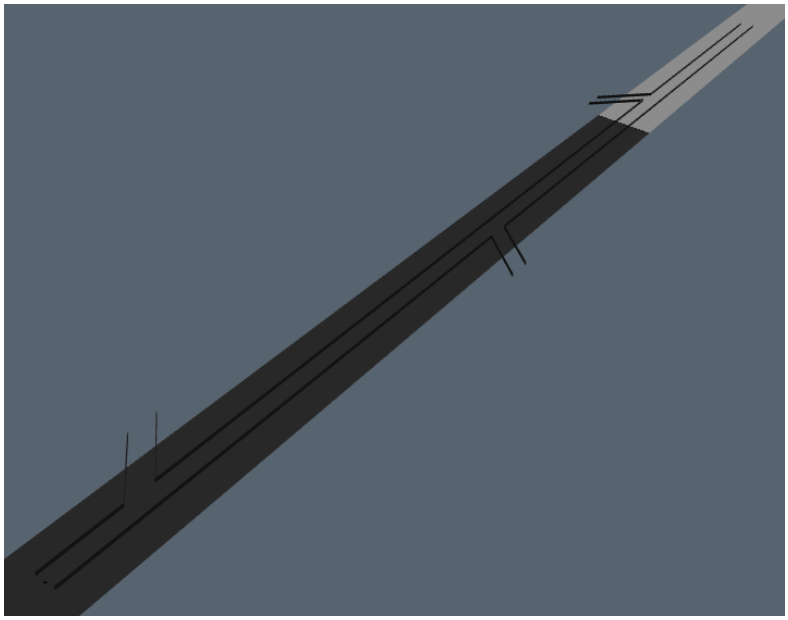


Figura 47. Túnel con galerías a izquierda y derecha.